Virtuelle Methoden für den Entwurf von fokussierten kombinatorischen Bibliotheken basierend auf Feature-Tree-Ähnlichkeit

Dissertation

zur Erlangung des akademischen Grades des

Doktors der Naturwissenschaften (Dr. rer. nat.)

an der Fakultät für Mathematik, Informatik und Naturwissenschaften

Fachbereich Informatik der

Universität Hamburg

vorgelegt von

Robert Fischer

aus Kronberg im Taunus

Hamburg, 2012

Genehmigt von der MIN-Fakulat Fachbereich Informatik der Universiat Hamburg auf Antrag von

Prof. Dr. Matthias Rarey (Erstgutachter/Doktorvater) und

Prof. Dr. Wolfgang Menzel (Zweitgutachter)

Hamburg, den 24.04.2012

Kurzfassung

Die kombinatorische Chemie erlaubt die systematische Synthese einer großen Anzahl von Molekülen in kurzer Zeit. Unter Verwendung von Synthesebausteinen und standardisierten Reaktionsschemata entstehen unterschiedliche Substanzen, die auf einer gemeinsamen Kernstruktur basieren. Diese Moleküle können anschließend mittels eines Hochdurchsatz-Screenings automatisiert auf ihre biologische Aktivität getestet werden. Aus Material- und Kostengründen kann jedoch lediglich eine begrenzte Anzahl von Bausteinen zur Synthese eingesetzt werden. Um die kombinatorische Chemie effektiv nutzen zu können, gilt es, die Bausteine so auszuwählen, dass möglichst viele Kombinationen der Bausteine zu Produkten führen, die den vorher definierten Kriterien entsprechen.

Wesentliche Kriterien für die Auswahl der Bausteine sind die physikochemischen Eigenschaften der resultierenden Moleküle, anhand derer zum Beispiel die Bioverfügbarkeit der Moleküle abgeschätzt werden kann. Weiterhin wird oftmals die Ähnlichkeit beziehungsweise Unähnlichkeit zu anderen Molekülen als Kriterium verwendet. Zum einen sollen die Produkte ähnlich zu biologisch aktiven Molekülen sein, zum anderen unähnlich zu unerwünschten Molekülen, um die Suche in eine bestimmte Richtung zu steuern. Soll ein möglichst großer Suchraum exploriert werden, sind Mechanismen erforderlich, die für eine möglichst diverse Auswahl der Bausteine sorgen.

Um diese unterschiedlichen Anforderungen zu erfüllen, wurde mit *LoFT* ein Verfahren für die multikriterielle Optimierung kombinatorischer Bibliotheken entwickelt. Durch die Verwendung des Feature-Tree-Deskriptors ist es möglich, die Ähnlichkeit der Produkte zu einem oder mehreren Anfragemolekülen zu betrachten, ohne die Produkte dafür explizit aus den jeweiligen Reaktanten zusammenzubauen. In Kombination mit Produkteigenschaften, die sich aus den Eigenschaften der Reaktanten ableiten lassen, erfolgt eine effiziente Bewertung der Produkte auf der Ebene der Baugruppen. Die simultane Betrachtung von Ähnlichkeit, Diversität und physikochemischen Eigenschaften führt zu Produkten, die den Anforderungen des jeweiligen Projektes gerecht werden.

Abstract

Combinatorial chemistry allows for the rapid and systematic synthesis of molecules using a unified reaction scheme. The resulting compounds have a common core, but differ in their so-called R-groups. They can be tested for biological activity using high throughput screening (HTS). Due to the large number of possible combinations, only a small number of building blocks can be selected for screening. In particular, for the effective usage of combinatorial chemistry, the building blocks have to be selected in a way, that the resulting products satisfy given criteria. Essential eligibility criteria of the building blocks are the physicochemical properties of the resulting products, for example to increase the chance for bioavailability. Additionally, molecular similarity is a key concept for selection. The resulting products should be similar to known actives and dissimilar to unwanted compounds, for example known inactive molecules. Furthermore, to explore a larger search space, mechanisms are required that lead to more diverse products.

In order to fulfill these requirements, the software tool *LoFT* was developed for the multi-objective optimization of combinatorial libraries. Incorporating the feature tree descriptor, the similarity of the products to one or more query molecules can be calculated without explicitly building them. In combination with product properties, which can be derived directly from the building blocks, an efficient product-based scoring can be obtained on reactant level. Thus, the simultaneous consideration of similarity, diversity and physicochemical properties leads to products, that satisfy the specific needs of the project.

Für Katja, Luca und Jona

Danksagung

Mein besonderer Dank gilt Professor Dr. Matthias Rarey für die interessante Aufgabenstellung und die exzellente Betreuung.

Das Projekt wurde in Kooperation mit Boehringer Ingelheim Pharma GmbH & Co. KG durchgeführt und von Boehringer Ingelheim Pharma GmbH & Co. KG finanziert. Den Initiatoren Dr. Uta Lessel und Dr. Herbert Köppen, die mit ihren Ideen die Arbeit entscheidend vorangebracht haben, möchte ich herzlich danken. Mein Dank gilt ebenfalls Dr. Alexander Weber und Dr. Bernd Wellenzohn, die bei den konstruktiven Treffen in Biberach ihren Teil dazu beigetragen haben, dass das Programm eine fortwährend positive Entwicklung genommen hat. Danke Uta, für die Mitarbeit an den Veröffentlichungen und die fortlaufende Erprobung und Überprüfung des Programms.

Ich danke der BioSolveIT GmbH, dass ich ihre Software-Bibliothek Flex* nutzen durfte. Insbesondere die Hilfe zu Beginn meiner Arbeit weiß ich zu schätzen.

Ein besonderer Dank gilt meinen Doktoranden-Kollegen am Zentrum für Bioinformatik für die ausgesprochen angenehme und sehr gute kollegiale Zusammenarbeit ebenso wie für die wissenschaftliche Diskussion untereinander. Danke Sascha, Tobias, Adrian, Matthias, Christian, Patrick, Jörg, Juri, Lennart, Axel, Jochen, Ingo, Andrea, Angela, Nadine, Katrin, Birte, Karen, Christin und Stefan.

Des Weiteren haben Jörgs Fragmentraum-Bibliothek und die von Patrick implementierten Funktionalitäten wesentlich dazu beigetragen, dass ich relativ schnell einen ersten Prototypen von *LoFT* erstellen konnte.
Die Arbeit im interdisziplinären Team mit Sascha, Adrian und Tobias an der Entwicklung von *NAOMI* hat mir sehr viel Freude bereitet und meine Kenntnisse in vielen Bereichen immens erweitert. Mein Dank gilt Tobias für

die gemeinsame Implementierung eines Moduls zur stochastischen Optimierung und der neuen Fragmentraum-Bibliothek im Rahmen der Entwicklung von *NAOMI*. Ebenso haben Matthias und Christian durch die Entwicklung des neuen 2D-Zeichners beziehungsweise SMARTS-Matchers die Integration von Funktionalitäten in mein Programm stark vereinfacht.

Bei der Beantwortung bürokratischer Fragen stand mir Melanie stets hilfreich zur Seite.

Meinen Korrekturlesern Katja, Uta, Sascha, Adrian, Matthias, Christian, Andrea, Angela, Susanne, Akira, Holger und Anne danke ich für die großzügige Hilfsbereitschaft und die wertvollen Anmerkungen und Verbesserungsvorschläge bei der Erstellung der vorliegenden Arbeit.

Vielen Dank auch an Herrn Prof. Dr. Wolfgang Mentzel für die Bereitschaft, ein Zweitgutachten zu erstellen.

Das Lächeln meiner beiden Söhnen Luca und Jona hat mich motiviert und gestärkt.

Mein größter Dank gilt jedoch dir, Katja, für deine uneingeschränkte und tatkräftige Unterstützung. Danke für deine unendliche Geduld.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

TABELLENVERZEICHNIS

1

Einleitung und Problemstellung

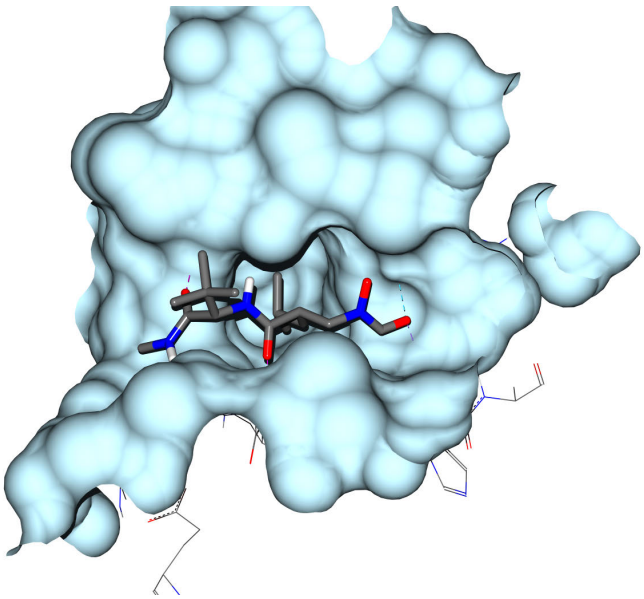


Abbildung 1.1: Protein-Ligand-Komplex - Die Abbildung zeigt die Metalloprotease MMP9 im Komplex mit einem Hydroxamat-Inhibitor (PDB-Code 1GKC [1]). Es wird vermutet, dass die Aktivität von MMP9 eine Hauptursache für die Entstehung von Herzinsuffizienz ist [2].

Proteine erfüllen vielfältige Aufgaben im Körper, unter anderem im Stoffwechsel und im Immunsystem. Viele Prozesse werden über die Wechselwirkung mit kleinen Molekülen, den Liganden, gekoppelt. Diese Interaktion basiert auf der chemischen und räumlichen Komplementarität von Protein und Ligand (siehe Abbildung 1.1). Dabei spielt die spezifische Anordnung der funktionellen Gruppen im Raum, Pharmakophor [4–6]

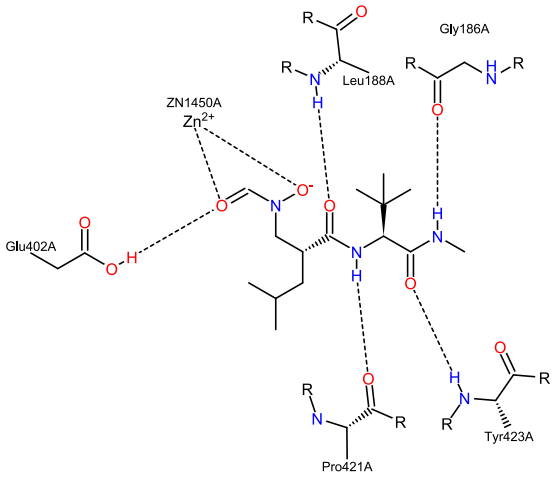


Abbildung 1.2: Schematische Darstellung der Wechselwirkungen im Protein-Ligand-Komplex - Die zweidimensionale Abbildung wurde mit Poseview [3] generiert und zeigt schematisch die vorhergesagten Interaktionen, die zur Komplexbildung von MMP9 und dem Hydroxamat-Inhibitor führen. Der Inhibitor bildet mehrere Wasserstoffbrücken sowie Wechselwirkungen zum Zink-Ion aus.

genannt, eine entscheidende Rolle. Der Pharmakophor kann aus wasserstoffbrückenbildenden (zwischen Wasserstoffbrückendonor und -akzeptor) sowie hydrophoben Teilen des Moleküls bestehen [7], welche mit dem aktiven Zentrum des Rezeptors, auch Bindungstasche genannt, in Wechselwirkung treten (siehe auch Abbildung 1.2). Diese Komplementarität wurde 1894 erstmals hypothetisch von Emil Fischer beschrieben [8] und wird als Schlüssel-Schloss-Prinzip bezeichnet.

Viele Arzneistoffe erzielen ihre Wirkung durch die Interaktion mit einem Makromolekül und verhindern die Bindung des natürlichen Substrates [10]. Motiviert durch das Ähnlichkeitsprinzip (*similar property principle*) [11], welches besagt, dass strukturell ähnliche Substanzen zu vergleichbaren physikochemischen und biologischen Eigenschaften tendieren, basierte die Entwicklung neuer Wirkstoffe in der Vergangenheit auf der seriellen und systematischen Modifikation chemischer Substanzen [12]. Anfang der 90er Jahre waren im Durchschnitt zwei bis drei Wochen notwendig, um ein Analogon, eine Substanz mit struktureller Ähnlichkeit zu einer anderen, zu synthetisieren. Dies wiederum führte zu Kosten in Höhe von 5000 $ - 7000 $ für die Synthese einer einzelnen Substanz [13].

Damit sich eine Substanz als Wirkstoff eignet, sind neben den Bindungseigenschaf-

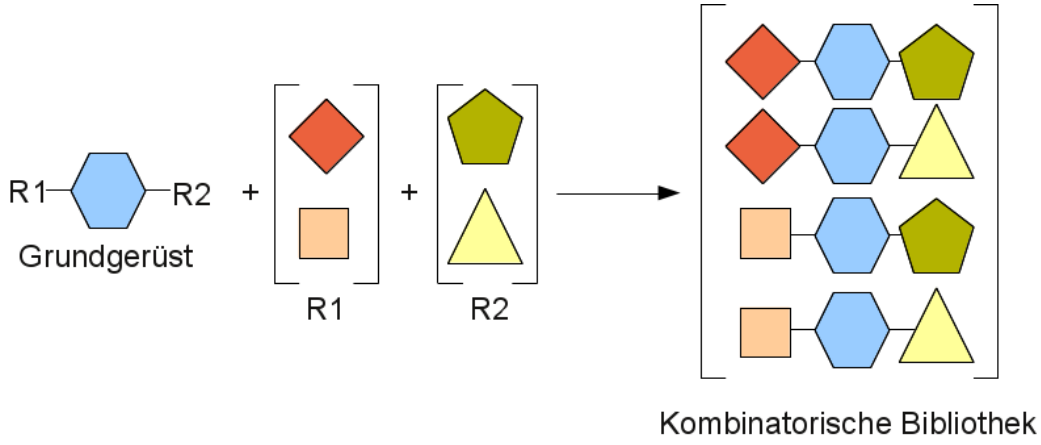


Abbildung 1.3: Schematische Darstellung einer kombinatorischen Bibliothek - Die kombinatorische Bibliothek ist eine Sammlung von Substanzen, die auf der systematischen Anwendung eines Syntheseschemas aus vordefinierten chemischen Bausteinen basiert. Dabei wird zumeist ein gemeinsames Grundgerüst verwendet, für dessen Substitutionsstellen (R1 und R2), eine Liste von Bausteinen (Reagenzien) zur Verfügung steht [9].

ten auch die pharmakokinetischen Eigenschaften[1] entscheidend. So muss der Wirkstoff beispielsweise seinen Wirkort erreichen können und darf nicht toxisch sein. Deshalb sollten die sogenannten ADMET-Eigenschaften (Absorption, Distribution, Metabolisierung, Exkretion und Toxizität) bei der Wirkstoffsuche so früh wie möglich Beachtung finden, um ungeeignete Substanzen auszuschließen [14].

Die Suche nach neuen Wirkstoffen ist dadurch ein komplexer, kostenintensiver und zeitaufwändiger Vorgang, der sich in mehrere Phasen unterteilen lässt und *In-vitro*-, *In-vivo*- und *In-silico*-Methoden umfasst [15]:

- Zunächst ist es erforderlich, die biologische Zielstruktur (*Target*) auszuwählen. Dafür wird auf Literaturrecherche, Genomforschung und vorhandene experimentelle Ergebnisse zurückgegriffen.

- In der Phase der Leitstrukturidentifizierung (*Lead Identification*) sollen die Strukturen ermittelt werden, die eine biologische Aktivität aufweisen. Aus diesen Treffern (*Hits*) werden dann vielversprechende Leitstrukturen ausgewählt. Diese die-

[1]Der Begriff Pharmakokinetik umfasst die Gesamtheit der Prozesse, denen ein Wirkstoff im Körper unterliegt.

nen als chemische Startpunkte für die weitere Optimierung und müssen die Synthese analoger Verbindungen erlauben.

- In der Phase der Leitstrukturoptimierung (*Lead Optimization*) sollen ausgewählte Leitstrukturen so modifiziert werden, dass die Wirkspezifität, Wirkhöhe und andere pharmakologische Parameter verbessert werden [16].

Die Substanzen werden in einem automatisierten Prozess auf ihre Eigenschaften getestet [17]. Die kombinatorische Chemie unterstützt diesen Vorgang, indem sie die parallele Synthese einer großen Anzahl strukturell verwandter Moleküle mit Hilfe des gleichen Reaktionsschemas ermöglicht (siehe Abbildung 1.3). Es entstehen Substanzen, die ein gemeinsames Grundgerüst enthalten, sich jedoch anhand ihrer Substituenten unterscheiden.

In den meisten Fällen übersteigt jedoch die Zahl der theoretisch möglichen Verbindungen die Synthese- oder Testkapazitäten. Deshalb muss ein geeignetes Auswahlverfahren gefunden werden, welches die Selektion der zu synthetisierenden Bausteine anhand bestimmter Kriterien ermöglicht. Dieser Prozess ist mittlerweile weitgehend computergestützt [18].

In der Vergangenheit wurden die Synthesebausteine hauptsächlich aufgrund von Diversität ausgesucht, um einen möglichst großen Teil des chemischen Raumes abzudecken [19–25]. Allerdings konnten diese unspezifische Bibliotheken die erwartete Trefferquote nicht erfüllen, beziehungsweise die gefundenen Treffer eigneten sich nicht als Leitstrukturen [23, 26]. Deshalb werden die kombinatorischen Bibliotheken inzwischen häufig auch hinsichtlich ihrer physikochemischen Eigenschaften optimiert. Außerdem werden fokussierte kombinatorische Bibliotheken zur Leitstrukturoptimierung eingesetzt. Für das Design solcher Bibliotheken werden spezielle Software-Werkzeuge verwendet, die eine simultane Optimierung unterschiedlicher, teils gegenläufiger Ziele, wie zum Beispiel strukturelle Ähnlichkeit, physikochemische und pharmakokinetische Eigenschaften [23], erlauben.

1.1 Ziel dieser Arbeit

Das Ziel der vorliegenden Arbeit war die Entwicklung eines Software-Werkzeuges zum Entwurf von fokussierten kombinatorischen Bibliotheken. In Zusammenarbeit mit Boehringer Ingelheim Pharma GmbH & Co. KG wurde ein Programm entwickelt, welches es

dem Nutzer ermöglicht, eine kombinatorische Bibliothek unter den folgenden Gesichtspunkten simultan zu optimieren:

- *Diversität:* Um einen möglichst großen chemischen Suchraum abzudecken, sollen die Substanzen auf einer möglichst diversen Auswahl der Bausteine beruhen.
- *Ähnlichkeit zu bekannten aktiven Substanzen:* Die entstehenden Moleküle sollen chemisch ähnlich zu bekannten biologisch aktiven Strukturen sein. Dies erlaubt zum Beispiel die rasche und kostengünstige Exploration von Leitstrukturen.
- *Unähnlichkeit zu unerwünschten Molekülen:* Je nach Problemstellung sollen die ausgewählten Substanzen möglichst unähnlich zu Molekülen sein, die inaktiv sind, patentrechtlich geschützt sind oder Nebenwirkungen haben.
- *Einhaltung eines vorgegebenen Profils von physikochemischen Eigenschaften:* Die Anforderung an ein bestimmtes physikochemisches Profil ergibt sich aus dem Wunsch nach Bioverfügbarkeit, sowie pharmakokinetischen und pharmakodynamischen Charakteristika, welche wiederum durch die physikochemischen Eigenschaften beeinflusst werden. Dabei ist eine möglichst geringe Abweichung von den angestrebten Zielgrößen gefordert, so dass die Bibliothek nicht einfach bezüglich ihrer Eigenschaften maximiert beziehungsweise minimiert werden kann.

Die Kriterien für die Auswahl richten sich nach den Anforderungen des Projektes und müssen sich für jede Bibliothek flexibel definieren lassen. Die Methode muss es zudem erlauben, die Optimierung durch Hinzunahme weiterer Kriterien iterativ zu verfeinern. Damit eine aufeinander abgestimmte Auswahl von Synthesebausteinen ausgewählt wird, ist es erforderlich, die Bibliothek in ihrer Gesamtheit zu optimieren. Denn wenn alle resultierenden Substanzen den geforderten Kriterien entsprechen, ist eine effiziente Umsetzung mittels kombinatorischer Chemie möglich. Qualitativ höherwertige Bibliotheken sind zudem zu erwarten, wenn bei der Optimierung die Eigenschaften der jeweiligen Produkte betrachtet werden, statt die ausgewählten Bausteine unabhängig voneinander zu bewerten [10, 27–29].

Die Neuartigkeit der in dieser Arbeit vorgestellten Methode besteht darin, dass zwar die Produkte bewertet werden, die Bewertung jedoch auf Ebene der Bausteine erfolgt. Im Gegensatz zu anderen existierenden Methoden wird dadurch die explizite Generierung aller Produkte der jeweiligen Subbibliothek vermieden. Um dies zu erreichen, findet

der Feature-Tree-Deskriptor [30] Anwendung, der die paarweise Ähnlichkeitsberechnung zweier Moleküle auf Ebene der molekularen Bausteine erlaubt [31].

Die im Rahmen dieser Arbeit vorgestellte Methode wurde bereits größtenteils publiziert [32, 33] und baut auf der Fragmentraum-Technologie [31, 34] sowie der in gemeinsamer Arbeit realisierten *NAOMI*-Bibliothek auf. Deren zugrundeliegendes Chemie-Modell wurde ebenfalls veröffentlicht [35].

1.2 Gliederung

Die folgenden Kapitel ordnen zunächst das in dieser Arbeit vorgestellte Verfahren in seinen Kontext ein. Kapitel 2 behandelt die Grundlagen des Entwurfs von kombinatorischen Bibliotheken. Die Verfahren, die als Basis für die entwickelte Anwendung dienen, werden in Kapitel 3 beschrieben. In Kapitel 4 wird das Chemie-Modell der *NAOMI*-Bibliothek vorgestellt und Modellierungs-Entscheidungen im Kontext dieser Arbeit erläutert. Kapitel 5 widmet sich dem Entwurf von fokussierten kombinatorischen Bibliotheken unter Berücksichtigung der Ähnlichkeit zu bekannten aktiven Molekülen. Das entwickelte Verfahren wird in Kapitel 6 anhand gängiger Arbeitsabläufe untersucht und validiert. Dabei wird auch der Einfluss der *NAOMI*-Bibliothek analysiert. In Kapitel 7 werden die Ergebnisse zusammengefasst. Außerdem werden mögliche Ansatzpunkte für eine Weiterführung dieser Arbeit skizziert. Die Anhänge enthalten ergänzende Beschreibungen. Detaillierte Anreicherungsdiagramme zur Validierung finden sich in Anhang A, die von der Anwendung unterstützten Dateiformate in Anhang B und eine Beschreibung der Implementierung in Anhang C. Anhang D enthält ein einführendes Tutorium in die Benutzung des Programms.

2

Design kombinatorischer Bibliotheken

Zu Beginn eines Projektes zur Wirkstoffsuche werden die Substanzen ausgewählt, die getestet werden sollen. In Frage kommen Naturstoffe, Substanzen von externen Anbietern, Substanzen aus früheren Projekten oder kombinatorische Bibliotheken. Kombinatorische Bibliotheken bestehen aus einer Menge von Verbindungen, die sich aus der Kombination von Synthesebausteinen mittels eines Reaktionsschemas ergeben. Bei der Planung kombinatorischer Bibliotheken (*Library Design*) stellt sich die Frage, welche Bausteine aus einer großen Menge von Bausteinen für die Synthese ausgewählt werden sollen. Diese Auswahl ist heute weitgehend computergestützt und ein wichtiger Bestandteil von Hochdurchsatzuntersuchungen.

Dieses Kapitel gibt eine Einführung in das Design kombinatorischer Bibliotheken. Für weiterführende Informationen sei auf die zahlreichen Übersichtsartikel und Bücher [9, 23, 25, 36–41] verwiesen, die zu diesem Thema bereits erschienen sind. Gerhard Klebe [42] bietet zudem eine umfassende Einführung in die Konzepte und Methoden des Wirkstoffentwurfs. Aktuelle Entwicklungen des computergestützten Wirkstoffentwurfs beschreibt Warr [43].

Da ein Großteil der Fachliteratur in englischer Sprache verfasst wurde, gibt es einige weit verbreitete und etablierte Ausdrücke, für die keine griffige deutsche Übersetzung existieren [18, 44]. In diesem Fall wird in der vorliegenden Arbeit der englische Ausdruck verwendet oder in Klammern hinter dem deutschen Begriff angegeben.

2.1 Hochdurchsatzuntersuchungen

Als Hochdurchsatzuntersuchungen (*High-Throughput-Experimentation*, HTE) wird eine Gruppe von Verfahren bezeichnet, die eine enorme Erhöhung des Durchsatzes bei der Herstellung und Testung von Stoffen für verschiedenste Anwendungen ermöglichen. Für die erfolgreiche Durchführung greifen vier unterschiedliche Teilgebiete ineinander [18]:

- Das Bibliotheksdesign
- Die Synthese der zu testenden Substanzen
- Die Testung (Screening)
- Automatisierung und Datenmanagement

Um die Projekte so effektiv und effizient wie möglich zu gestalten, werden automatisierte und robotergestützte Verfahren eingesetzt. Entscheidend ist, dass die einzelnen Verfahren gut aufeinander abgestimmt sind und eine effiziente Datenverwaltung ermöglichen.

2.1.1 Kombinatorische Synthese

Die kombinatorische Synthese ist eines der möglichen Syntheseverfahren die bei Hochdurchsatzuntersuchungen zum Einsatz kommen. Während bei der klassischen Synthese ein bestimmtes Zielmolekül hergestellt werden soll, zielt die kombinatorische Synthese auf die Erzeugung einer Gruppe von Substanzen ab. In einem mehrstufigen Prozess werden nach festgelegten Synthesevorschriften Synthesebausteine eines definierten Types miteinander umgesetzt. Das divergente Verfahren resultiert somit in einer Produktmatrix (siehe Abbildung 2.1). Dafür ist es notwendig, dass die Reaktionsbedingungen und Synthesebausteine so gewählt werden, dass alle Kombinationen zuverlässig reagieren [16]. Denn nicht jede Substanz ist über die kombinatorische Chemie zugänglich.

Die kombinatorische Synthese kann sowohl in Lösung als auch an fester Phase erfolgen. Bei der Synthese in Lösung erlauben Pipetierautomaten die Dosierung der Reagenzien. Wenn eine Aufreinigung vermieden werden soll, ist dieses Verfahren jedoch auf ein bis zwei Syntheseschritte beschränkt. Bei der Synthese an fester Phase wird ein polymerer Träger, vorzugsweise ein Harz verwendet. Die Harzkügelchen sind über einen sogenannten Linker an das Molekül gebunden. Nach dem vollständigem Aufbau des

Moleküls wird der Linker abgespalten. Die Wahl des Linkers hat dabei Einfluss auf die Syntheseplanung, da er die Abspaltungsbedingungen und die endständige funktionale Gruppe, die am Produkt nach der Abspaltung erhalten bleibt, definiert [16].

Für die Umsetzung der Synthese stehen verschiedene Verfahren zur Verfügung. Bei der *Parallelsynthese* entsteht in jedem Reaktionsgefäß ein anderes Produkt. Für die effiziente Umsetzung der Substanzen werden Syntheseautomaten verwendet. Das *Mix-and-Split*-Verfahren eignet sich für die Synthese großer Bibliotheken. Es sind weniger Syntheseschritte notwendig, da die Zwischenstufen als Mischungen zur Reaktion gebracht werden. Schueth [18] sowie Frobel und Krämer [16] erklären den Unterschied zwischen den Vorgehensweisen anhand einer dreistufigen Synthese mit jeweils drei Synthesebausteinen. Bei der herkömmlichen und der parallelen Synthese sind 81 Syntheseschritte für die 27 Produkte notwendig. Beim Mix-and-Split-Verfahren hingegen sind lediglich 9 Syntheseschritte notwendig. Die entstandenen Produkte liegen dann jedoch in zum Teil recht komplexen Mischungen vor.

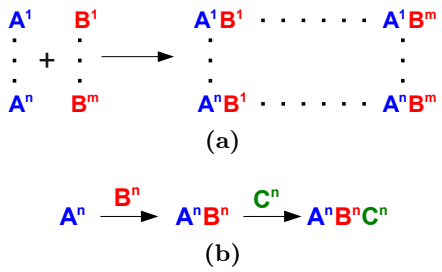


Abbildung 2.1: Kombinatorische Synthese - (a) Synthesebausteine vom Typ A werden mit Bausteinen von Typ B umgesetzt, so dass alle möglichen Produkte entstehen. (b) Bei einer mehrstufigen Synthese entsteht schnell eine große Zahl von Produkten. Für die dreistufige Synthese mit $n = 10$ ergeben sich in diesem Beispiel 1000 Produkte, für $n = 100$ ergeben sich bereits eine Million Produkte. Vergleiche auch Balkenhohl et al. [45]

2.1.2 Experimentelles Screening

Für die Testung der Substanzen auf biologische Aktivität können Bindungsassays, Enzyminhibitionsassays oder zellbasierte Bioassays verwendet werden. Bei diesen Testsystemen handelt es sich um spezifische Nachweismethoden nach einem standardisierten Reaktionsverfahren. Viele dieser Testverfahren lassen sich dadurch in automatisierten Verfahren anwenden. Bei einer solchen automatischen Testung mit hohem Durchsatz wird von einem Hochdurchsatz-Screening (*High-Throughput-Screening*, HTS) gesprochen.

Prinzipiell wird zwischen zellfreien und zellbasierten Assays unterschieden. Während erstere die Aktivität des Moleküls in Bezug auf ein reines Zielprotein testen, indem die räumliche Nähe oder die reduzierte Beweglichkeit des gebunden Moleküls gemessen wird, wird bei zellbasierten Assays die biologische Antwort der Zelle analysiert. Hierbei gibt es weniger direkte Detektionsmechanismen, es werden jedoch auch Informationen über Cytotoxizität und Bioverfügbarkeit geliefert [18].

Die Tests werden ausgewertet und Substanzen, bei denen die Messwerte über den Schwellenwerten liegen, als Treffer (*Hits*) klassifiziert. Viele dieser Treffer eignen sich jedoch nicht als Leitstruktur oder gar als Wirkstoff. Aus diesem Grund werden mittlerweile ADMET-Tests, die sich ebenfalls automatisieren lassen, bereits in frühen Phasen des Wirkstoffentwurfs eingesetzt.

2.1.3 Virtuelles Screening

Das virtuelle Screening (VS) hat sich als eine kostengünstige und schnelle Alternative zum experimentellen Screening etabliert [46, 47]. Zusätzlich bietet es den Vorteil, dass Substanzen untersucht werden können, die physisch nicht vorhanden sind und erst bei Bedarf gekauft oder synthetisiert werden. Grundlegend wird zwischen strukturbasiertem und ligandbasiertem Screening unterschieden [48]. Ersteres basiert auf der 3D-Struktur des Zielproteins. Das Docking (siehe zum Beispiel Muegge und Rarey [49]) bezeichnet das Einpassen eines Liganden in die Bindetasche. Beim ligandbasierten Screening dagegen dienen biologisch aktive Substanzen als Referenz- oder Anfragemolekül für die ähnlichkeitsbasierte Suche. Dieser Ansatz wird insbesondere verwendet, wenn die Struktur des Proteins nicht bekannt ist.

2.2 Rahmenbedingungen bei der Planung kombinatorischer Bibliotheken

Virtuelle kombinatorische Bibliotheken ergeben sich durch die Kombination verschiedener Reaktanten in einem einheitlichen Reaktionsschema. Dabei wird oftmals eine Markush-Struktur verwendet, bestehend aus einem Grundgerüst mit expliziten Punkten, an die Reagenzien angebaut werden können [51]. Das Grundgerüst ist das Kernstück eines Moleküls, das allen Verbindungen einer kombinatorischen Bibliothek gemein ist. So kann die allgemeine Struktur einer Bibliothek durch das Grundgerüst dargestellt

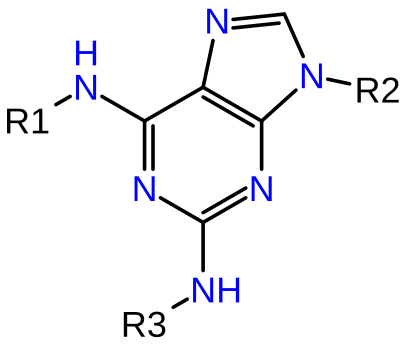


Abbildung 2.2: Generische Struktur einer Purin-Bibliothek - Die Abbildung zeigt das Grundgerüst einer Purin-Bibliothek mit drei Substitutionsstellen, repräsentiert durch R1, R2 und R3 [50].

werden, wobei die Positionen, an denen sich die variablen Reagenzien befinden, gekennzeichnet sind [44]. Diese Substitutionsstellen beschreiben die offenen Valenzen des Grundgerüstes. Exemplarisch zeigt Abbildung 2.2 ein Purin-Grundgerüst mit drei Substitutionsstellen. Die bei der Synthese eingesetzten Reagenzien führen in den Produkten zu R-Gruppen beziehungsweise Substituenten. Aus dieser Strukturklasse resultierende Produkte unterscheiden sich folglich nur anhand ihrer Substituenten.

Das Design kombinatorischer Bibliotheken ist ein Auswahlprozess mit dem Bestreben, die Anzahl der Produkte mit den gewünschten Eigenschaften zu maximieren und gleichzeitig die Anzahl derer mit unerwünschten Eigenschaften zu minimieren [52]. Es werden zwei Designziele unterschieden, der Entwurf von diversen und fokussierten Bibliotheken: [52–54]

- *Design diverser Bibliotheken (diverse/general libraries):* In der Phase der Leitstrukturidentifizierung werden möglichst diverse Bibliotheken generiert, um einen möglichst großen chemischen Suchraum abzudecken. Werden vielversprechende Kandidaten detektiert, werden diese im fokussierten Design weiter optimiert. In der Vergangenheit war Diversität das alleinige Designziel. Um jedoch Treffer zu

finden, die sich als Leitstrukturen eignen, werden die Reagenzien auch anhand physikochemischer Eigenschaften ausgewählt [55].

- *Design fokussierter Bibliotheken (focussed/directed libraries):* In der Phase der Leitstrukturoptimierung wird angestrebt, so viel Wissen wie möglich über das Zielprotein in das Design einfließen zu lassen. Ligandbasierte Verfahren können angewandt werden, um chemisch ähnliche Moleküle zur Leitstruktur zu generieren. Ist die Struktur des Proteins oder gar des Komplexes bekannt, können strukturbasierte Verfahren Anwendung finden.

Tabelle 2.1 findet sich im 1999 erschienenen Buchkapitel von Dominika Tiebes [56] und fasst die unterschiedlichen Designansätze zusammen.

	Leitstrukturidentifizierung	**Leitstrukturoptimierung**
Bibliothekstyp	generisch	fokussiert
Bibliotheksgröße	groß, mehr als 10 000 Verbindungen, weniger als 1 mg pro Molekül	moderat, weniger als 10 000 Verbindungen, mehr als 1 mg pro Molekül
Strukturziel	unspezifisch	spezifisch (leitstrukturorientiert)
Strukturelle Diversität	größtmöglich	beschränkt
Bausteine	jeder diverse	spezifisch in Bezug auf die Retrosynthese
Synthesestrategie	flexibel	genau definiert
Screeningziel	viele biologische Ziele	ein biologisches Ziel oder eine Zielklasse

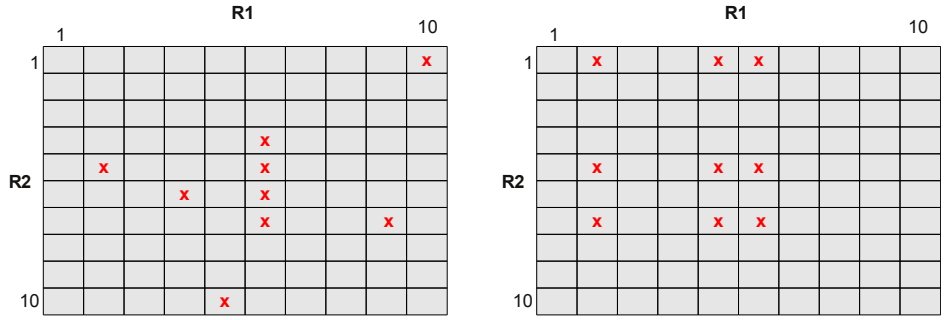
Tabelle 2.1: Bibliotheksdesign im Falle von Leitstrukturidentifizierung und Leitstrukturoptimierung. Adaptiert von Tiebes [56].

Neben der Frage, wie die Synthesebausteine und Regeln im Programm repräsentiert werden sollen, sind bei der Modellierung eines Programmes zum Entwurf von kombinatorischen Bibliotheken einige grundlegende Design-Entscheidungen zu treffen. Diese Entscheidungen sind unter anderem abhängig von der Art der Anwendung und Zielsetzung des Designs:

- Soll eine Subbibliothek in ihrer Gesamtheit optimiert oder sollen die besten Produkte in Bezug auf die Bewertungsfunktion gewählt werden?

- Sollen die Reagenzien oder die Produkte bewertet werden?
- Mit welchem Verfahren soll der chemische Raum durchsucht werden?
- Welche Kriterien fließen in die Bewertung ein?

2.2.1 Format



(a) Sparse Array **(b)** Full Array

Abbildung 2.3: Kombinatorische Untermengen - Auswahl einer Menge von Produkten (Sparse Array) oder einer Subbibliothek (Full Array).

Eine kombinatorische Bibliothek besteht bei R Komponenten und n_i Synthesebausteinen aus $n = \prod_{i=1}^{R} n_i$ möglichen Produkten.

Die Auswahl einer beliebigen Untermenge k aus n Substanzen wird als *Sparse-Array* bezeichnet. Für ein solches Sparse-Array existieren $\binom{n}{k} = \frac{n!}{(n-k)!\cdot k!}$ Möglichkeiten. Die Substanzen repräsentieren dabei nicht unbedingt alle möglichen Produkte, die aus den gewählten Synthesebausteinen generiert werden können. Für die Auswahl der k Substanzen werden Verfahren zum sogenannten "Rosinenpicken" (*Cherry-Picking*) verwendet.

Ein *Full-Array* bezeichnet dagegen eine Untermenge von Substanzen, die aus allen möglichen Kombinationen der gewählten Synthesebausteine gebildet wird. Über die Anzahl der gewünschten Reagenzien für jeden Substituenten definiert sich dabei die Größe der fokussierten Bibliothek ($\prod_{i=1}^{R} k_i$). Für jeden Substituenten i gibt es $\binom{n_i}{k_i}$ Möglichkeiten, k_i aus einer Menge von n_i Reagenzien zu wählen. Die Anzahl der möglichen Teilbibliotheken ergibt sich folglich aus der Formel 2.1:

$$\prod_{i=1}^{R} \binom{n_i}{k_i} \tag{2.1}$$

Das Problem, eine k-großen Untermenge auszuwählen, ist NP-vollständig und die Kardinalität des Raumes ist selbst in sehr konservativen Designszenarien enorm [20]. Bereits für die Auswahl von 10 aus 100 Reagenzien ergeben sich $1,73 \cdot 10^{13}$ Möglichkeiten. Sollen bei zwei Substitutionsstellen jeweils die 10 besten aus 100 Reagenzien gewählt werden, ergeben sich bereits ungefähr $3 \cdot 10^{26}$ Möglichkeiten. Die Schwierigkeit liegt somit in der Aufgabe, eine Subbibliothek zu generieren, bei der die betrachteten Kriterien für alle Produkte bestmöglich erfüllt sind.

Die meisten kombinatorischen Bibliotheken werden im Full-Array Format synthetisiert [57]. Full-Arrays sind einfacher zu planen und lassen sich mit einem Syntheseroboter synthetisieren. Dadurch kann es ratsam sein, die Synthese nicht nur anhand der Moleküle mit den besten Eigenschaften zu planen (Cherry-Picking), sondern gegebenenfalls etwas ungünstigere Eigenschaften in Kauf zu nehmen, die dann jedoch für mehr Produkte gelten.

2.2.2 Reaktant- und produktbasierter Ansatz

Beim Design können zwei unterschiedliche Strategien verfolgt werden, der reaktantbasierte und der produktbasierte Ansatz [53, 58, 59]. Im reaktantbasierten Ansatz werden die Bausteine anhand ihrer Eigenschaften ausgewählt, ohne die Eigenschaften der resultierenden Produkte zu berücksichtigen. Im produktbasierten Ansatz werden dagegen die Produkteigenschaften betrachtet. Um die Produkte generieren und bewerten zu können, ist in der Regel eine Enumeration der virtuellen Bibliothek notwendig. Dieser Vorgang ist zwar aufwändiger, jedoch vielversprechender in Bezug auf die Optimierung der Eigenschaften und Diversität der gesamten Bibliothek [10, 27–29].

2.2.3 Optimierungsverfahren

Die kombinatorische Natur des Problems erlaubt es nicht, alle möglichen Produkte und Subbibliotheken zu enumerieren und zu bewerten. Für die Exploration des chemischen Raum werden deshalb heuristische Verfahren genutzt. Häufige Anwendung finden stochastische Optimierungsmethoden wie die Simulierte Abkühlung (*Simulated Annealing*, SA) [60] und Evolutionäre Algorithmen (*Evolutionary Algorithm*, EA), wie der Genetische Algorithmus (*Genetic Algorithm*, GA) [10]. Die genannten Verfahren erlauben das erfolgreiche Design von Bibliotheken, obwohl lediglich ein kleiner Teil des Suchraums betrachtet wird [23]. Auf das Verfahren der Simulierten Abkühlung wird in Kapitel 5.5

noch einmal näher eingegangen. Bei Verwendung eines Genetischen Algorithmus wird eine Population anhand genetischer Operatoren (Mutation und Rekombination) verändert und mittels Selektion optimiert. Ein einzelnes Chromosom kodiert hierbei entweder ein Molekül oder eine Bibliothek [61].

2.2.4 Bewertungsfunktionen

Um die Güte einer fokussierten Bibliothek zu bestimmen, wird eine Fitness- oder Bewertungsfunktion benötigt. In der Vergangenheit wurden Bibliotheken zunächst auf ein einzelnes Kriterium hin optimiert. Implizit gehen die Methoden davon aus, dass das Optimum eines einzelnen Kriteriums auch dem Optimum der anderen Kriterien entspricht. Gerade im Kontext des Wirkstoffdesigns können jedoch die Optima einzelner Kriterien substantiell variieren [63].

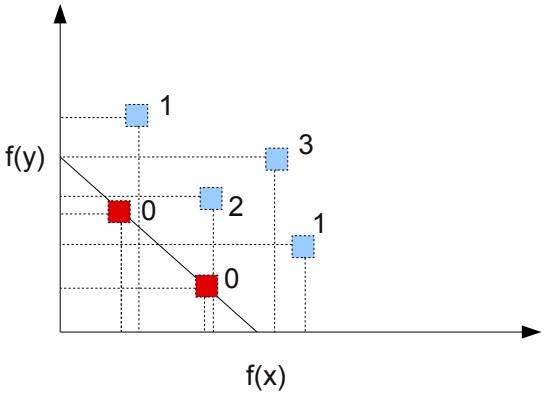


Abbildung 2.4: Pareto-Optimierung - Die Abbildung zeigt eine Menge von Lösungen bei der Minimierung zweier Kriterien x und y. Die Zahl neben der Lösung gibt die Anzahl der dominierenden Lösungen an. Die rot markierten Lösungen bilden die Pareto-Front. (Vergleiche auch Gillet [26] oder Nicolaou und Kannas [62]).

Mittlerweile werden Bibliotheken simultan auf mehrere, oftmals gegenläufige, Kriterien hin optimiert. Insofern muss ein Kompromiss gefunden werden: Es wird nach Lösungen gesucht, die in Bezug auf alle Kriterien akzeptabel sind, auch wenn sie, werden die Kriterien einzeln betrachtet, vielleicht suboptimal sind [26].

Um einen solchen Kompromiss zu finden, können die Kriterien in einer gewichteten Bewertungsfunktion kombiniert werden. Eine solche Bewertungsfunktion hat die Form $\sum_i^n \omega_i s_i$, wobei bei n Eigenschaften ω_i das Gewicht und s_i die Bewertung des Kriteriums i ist. Der Nachteil besteht darin, dass der Benutzer die Gewichtung selbst festlegen muss. Ein anderer Ansatz ist die Suche nach dem Pareto-Optimum. Hierbei werden die Ziele nicht gewichtet. Stattdessen werden die Lösungen nach ihrer Dominanz gegenüber anderen Lösungen in den einzelnen Optimierungskriterien sortiert. Bei einer pareto-optimalen Lösung kann die Bibliothek in Bezug auf ein Kriterium nur verbessert werden,

wenn sich die Bewertung eines anderen Kriteriums verschlechtert. Im besten Fall gibt es somit eine Lösung, die alle anderen dominiert. Falls eine solche Lösung nicht existiert, gibt es mehrere nicht-dominierte Lösungen. Das führt allerdings zu dem Problem, dass es zu viele gleichwertige Lösungen geben kann (Pareto-Front, siehe Abbildung 2.4). In diesem Fall muss Expertenwissen einfließen und eine Gewichtung erfolgen. Eine weitere Möglichkeit zur Bewertung ist ein Wünschbarkeitsindex [64–66]. Für jedes Merkmal werden Grenzwerte festgelegt und mittels einer Wünschbarkeitsfunktion (siehe Kapitel 5.6) in das Intervall [0,1] transformiert. Der Wünschbarkeitsindex wiederum fasst die Wünschbarkeiten, beispielsweise durch Berechnung des geometrischen Mittelwertes, in einem einzelnen Wert zusammen.

2.2.5 Deskriptoren

Für die jeweilige Anwendung müssen die Moleküle auf geeignete Weise dargestellt werden. Eine solche mathematische Beschreibung wird als Deskriptor bezeichnet. Deskriptoren werden anhand der zu kodierenden Informationen in drei Klassen unterteilt:

- **1D-Deskriptoren** beschreiben eine makroskopische Eigenschaft des Moleküls und fassen diese als einen einzelnen Wert zusammen. Prinzipiell kann jede molekulare Eigenschaft als Deskriptor verwendet werden. Beispiele sind die Molekülmasse oder die Anzahl der Schweratome. 1D-Deskriptoren werden häufig zum Abschätzen der ADMET-Eigenschaften (siehe Kapitel 2.3.1) eingesetzt, haben jedoch für den jeweiligen Medizinalchemiker, Biochemiker oder Pharmakologen im entsprechenden Kontext unter Umständen eine unterschiedliche Bedeutung [67]. Sind die Eigenschaften additiv über die Baugruppen eines Moleküls, erlauben sie das effiziente Filtern von Substanzen anhand der kombinierten Eigenschaften der jeweiligen Bausteine [68–70].

- **2D-Deskriptoren** beschreiben die Konnektivität der Atome und somit den Molekülgraphen. Sie enthalten topologische Informationen, die 3D-Information wird jedoch verworfen oder auf 2D projiziert. Ein 2D-Deskriptor ist zumeist eine vektorielle Repräsentation des Moleküls. So beschreiben *Structural Keys* die auftretenden Bausteine in einem Molekül (beispielsweise MACCS Keys [71]). *Hashed Fingerprints* hingegen kodieren die im Molekül vorkommenden Pfade einer bestimmten Länge (beispielsweise Daylight Fingerprints [72]). *Atom Environment Descriptors*

[73, 74] wiederum beschreiben die Umgebung der Atome (zum Beispiel *Extended/Functional Connectivity Fingerprints* (ECFP, FCFP) [75, 76]). Zum Vergleich kann beispielsweise der Tanimoto-Koeffizient verwendet werden, welcher bei der Evaluierung auf chemischen Daten gute Ergebnisse erzielte [77, 78]. Eine detaillierte Übersicht über gängige Vergleichsmaße für Vektorrepräsentationen findet sich bei Willett et al. [77].

- **3D-Deskriptoren** beschreiben die Molekülgestalt, indem die räumlichen Abstände der Atome zueinander betrachtet werden. Ein Beispiel ist die molekulare Oberfläche.

Da 3D-Deskriptoren den Konformationsraum der Moleküle in Betracht ziehen müssen, haben sich 2D-Deskriptoren in der ligandbasierten Ähnlichkeitssuche als vorteilhaft erwiesen [79–83]. Ursprünglich wurden 2D-Deskriptoren jedoch zur Substruktursuche entwickelt. Sie eignen sich daher vor allem zur Identifikation analoger Strukturen und weniger zur Identifikation von Substanzen mit der gleichen Aktivität, die auf anderen Strukturklassen beruhen [83]. Unter Verwendung von 2D-Deskriptoren werden oftmals Moleküle ausgewählt, die dem Anfragemolekül strukturell sehr ähnlich sind.

Eine besondere Form der 2D-Deskriptoren sind reduzierte Graphen [83–87], die den molekularen Graph durch eine Baumstruktur abstrahieren. Dadurch bleibt die topologische Anordnung der funktionellen Gruppen erhalten, ohne dass es erforderlich ist, die möglichen Konformationen der Moleküle zu betrachten. Sie werden zwischen 2D- und 3D-Deskriptoren eingeordnet [30] und finden insbesondere Anwendung, wenn die relative Anordnung der funktionellen Gruppen wichtiger ist als exakte Substruktur-Überlagerungen. Dies ist bei der Ähnlichkeitssuche und der Clusteranalyse von chemischen Strukturen der Fall. Insofern ergänzen reduzierte Graphen die bestehenden 2D- und 3D-Ansätze [88], da sie sich auch für den Grundgerüstwechsel (*Scaffold hopping*) eignen [88, 89]. Ausgehend von einer bereits bekannten aktiven Struktur, ermöglichen sie es, neue Verbindungen zu finden, die ein anderes Grundgerüst besitzen [90].

Für die Berechnung der paarweisen Ähnlichkeit zwischen zwei reduzierten Graphen gibt es mehrere Ansätze, unter anderem die Verwendung von Pseudo-SMILES [83] und die Edit-Distanz[1] [88, 91]. Der Feature-Tree-Deskriptor ist ebenfalls ein reduzierter

[1] Die Anzahl der Operationen, die notwendig ist, um eine Zeichenkette in eine andere zu transformieren.

Graph. Die paarweise Ähnlichkeit zweier Feature-Trees wird durch die Berechnung des maximal-gewichteten bipartiten Matchings bestimmt (siehe Kapitel 3.2).

Die Wahl von Deskriptor und Ähnlichkeitsmaß beeinflusst die Qualität und Geschwindigkeit der Berechnung. Sie ist aber auch bedingt durch den Anwendungsfall und die Art der verfügbaren Informationen. So kann es in vielen Fällen von Vorteil sein, 2D- und 3D-Deskriptoren gemeinsam zu nutzen [92, 93]. Eine detaillierte Übersicht über gängige Deskriptoren findet sich bei Gasteiger und Engel [94], Todescini und Consonni [95] sowie Mannhold, Kubinyi und Folkers [67].

2.3 Auswahlkriterien

Für die Generierung von Bibliotheken, die den Anforderungen des Projektes entsprechen, ist die Auswahl geeigneter Auswahlkriterien entscheidend. Vor allem physikochemische Eigenschaften sowie ligand- beziehungsweise strukturbasierte Verfahren finden Anwendung, um die Wahrscheinlichkeit zu erhöhen, dass die Bibliothek aktive Substanzen enthält.

Dabei ist je nach Projektstatus und Anforderungen die Diversität der Synthesebausteine und somit auch der Produkte ein wichtiger Parameter. Ebenso können Kriterien wie der Preis oder die Verfügbarkeit der Synthesebausteine von Interesse sein.

2.3.1 Physikochemische Eigenschaften

Begründet auf der Annahme, dass biologisch und pharmakologisch aktive Moleküle nicht gleichmäßig über den gesamten chemischen Raum verteilt sind, lassen sich wahrscheinlichere Regionen mittels einiger weniger physikochemischer Eigenschaften charakterisieren [96–98]. Um Moleküle aus solchen Subräumen auswählen und um insbesondere die ADMET-Eigenschaften der Substanzen abschätzen zu können, werden Regeln und Kriterien aufgestellt. Dies wird durchaus kritisch beurteilt [99–102], da sich insbesondere die Vorhersage von Lipophilie als schwierig erweist [102].

Beim Design kombinatorischer Bibliotheken ist es dennoch notwendig, die Synthesebausteine anhand geeigneter Eigenschaften auszuwählen. Insbesondere in der Phase der Leitstrukturidentifizierung sind Bibliotheken vorzuziehen, die in Substanzen mit geringerer Komplexität resultieren, welche wiederum als Leitstrukturen weiter optimiert

werden können [103, 104]. Goodnow et al. [105] merken an, dass ohne geeignete Auswahlkriterien gerade bei der Feststoffsynthese oftmals Strukturen synthetisiert werden, die zu lipophil, schwer und komplex für die Leitstrukturoptimierung sind.

Die bekannteste und am weitesten verbreitete Regel ist die *Rule-of-Five* [106], die Lipinski und Mitarbeiter 1997 aufstellten. Sie besagt, dass die Wahrscheinlichkeit der oralen Bioverfügbarkeit[1] signifikant sinkt, wenn zwei oder mehr der folgenden Filterkriterien verletzt sind:

- Nicht mehr als fünf Wasserstoffbrückendonoren
- Nicht mehr als zehn Wasserstoffbrückenakzeptoren
- Eine relative Molekülmasse von maximal 500 Dalton
- Der Wert des berechneten Logarithmus des Octanol-Wasser-Koeffizienten (clogP) beträgt nicht mehr als fünf Einheiten

Die Regel wurde 1999 von Ghose et al. [107] für das Bibliotheksdesign erweitert. Weitere Beispiele sind die Regeln für die Auswahl von Substanzen für die Leitstrukturoptimierung [104, 108] und die *Rule-of-Three* für die Auswahl von Fragmenten für experimentelle Untersuchungen [109]. Norinder und Haeberlein [110] stellen die Regel auf, dass die Summe der Sauerstoff- und Stickstoffatome eines Moleküls maximal fünf betragen sollte, damit die Substanz die Blut-Hirn-Schranke passieren kann. Zudem wurde beobachtet, dass eine Verringerung der Anzahl der Wasserstoffbrückendonoren die orale Aufnahme verbessern kann [111]. Weiterhin korreliert die polare Oberfläche (*polar surface area, PSA*) mit dem passiven Transport von Molekülen durch die Membran. [112–114] Veber et al. [115] sehen eine polare Oberfläche kleiner als 140 Angström und zehn oder weniger rotierbare Bindungen als Indikator für eine wahrscheinliche orale Bioverfügbarkeit in der Ratte an. Pickett et al. [116] betrachten dagegen LogP, Molekulargewicht und polare Oberfläche, um kombinatorische Bibliotheken zu generieren, deren Produkte verbesserte Absorptionseigenschaften besitzen.

[1] Die orale biologische Verfügbarkeit ist eine Messgröße für den Anteil eines Wirkstoffes, der nach der oralen Gabe am Wirkort zur Verfügung steht.

2.3.2 Ligand- und strukturbasierte Verfahren

Je nach Art der zur Verfügung stehenden Informationen über das Zielprotein werden neben physikochemischen Eigenschaften auch ligand- und strukturbasierte Verfahren eingesetzt. Die meisten Programme für das fokussierte Bibliotheksdesign integrieren eines oder mehrere dieser Verfahren. Die Art der gewählten Deskriptoren entscheidet dabei maßgeblich über die Laufzeit des Programms. Insbesondere beim produktbasierten Design müssen die Produkte für die Bewertung normalerweise enumeriert und die betreffenden Deskriptoren generiert und angewandt werden. Von Vorteil sind Verfahren, die speziell auf den Anwendungsfall kombinatorischer Bibliotheken zugeschnitten sind. So wurde beispielsweise für das molekulare Docking-Verfahren FlexX [117] eine effiziente Erweiterung für das Docking von kombinatorischen Bibliotheken entwickelt [118].

2.3.3 Diversität der ausgewählten Bausteine

Für den Begriff der "Molekularen Diversität" existiert keine objektive Definition [22]. Die bestehenden Verfahren für den Entwurf diverser Bibliotheken haben ihren Ursprung in Methoden wie der Substruktur- und Ähnlichkeitssuche, Clusteranalyse und QSAR und basieren somit ebenfalls auf dem Ähnlichkeitsprinzip [11]. Einen umfassenden Überblick geben die Übersichtsartikel [19–22, 119].

Eine der ersten Methoden, die für das Design diverser Bibliotheken angewandt wurde, ist die Gruppierung der Substanzen anhand ihrer Deskriptoren durch Clusteranalyse-Verfahren und die damit verbundene Auswahl eines Repräsentanten jedes Clusters [23]. Des Weiteren werden für die Generierung von diversen Bibliotheken auch Simulierte Abkühlung, Genetische Algorithmen und Neuronale Netze genutzt [22]. Gillet und Willett [119] unterteilen die gängigen Verfahren in drei Kategorien:

- *Zellbasierte Methoden* bestimmen absolute Positionen im Raum. Die Werte der betrachteten Eigenschaften werden in Bereiche aufgeteilt. Das kombinatorische Produkt aller möglichen Teilbereiche ergibt ein k-dimensionales Gitter. Die Moleküle werden bezüglich ihres Profil einer Zelle zugeteilt. Anschließend wird aus jeder Zellen eine Substanz selektiert [29].

- *Clusterbasierte Methoden* haben zum Ziel, Moleküle so in Gruppen (Cluster) einzuteilen, dass die Moleküle eines Clusters ähnlich, Moleküle unterschiedlicher Cluster jedoch unähnlich sind. Es wird zwischen hierarchischen und nicht-hierarchischen Ansätzen unterschieden. Hierarchisch-agglomerative Verfahren weisen alle Objekte zunächst einem eigenen Cluster zu. Die Cluster beziehungsweise Objekte mit der geringsten Distanz werden zusammengeführt, bis die maximal erlaubte Distanz erreicht ist. Die Distanz zweier Cluster kann dabei unter anderem auf dem minimalen (*Single Linkage*) oder maximalen (*Complete Linkage*) Abstand zweier beliebiger Punkte der jeweiligen Cluster beruhen. Ebenso kann die durchschnittliche Distanz aller Punkte zweier Cluster (*Average Linkage*) oder die Zunahme der Varianz beim Vereinigen zweier Cluster (Wards Kriterium [120]) als Kriterium dienen. Zu den nicht-hierarchischen Verfahren zählen Austauschverfahren wie k-Means [121] oder k-Medoid [122] sowie Nächste-Nachbarn-Verfahren wie Jarvis-Patrick [123]. Insbesondere die Methode von Ward [120] beziehungsweise das k-Means-Verfahren [121] finden in der Chemieinformatik Anwendung und haben das bis dahin populäre Jarvis-Patrick [123] abgelöst [23]. Einen Überblick über Clusteranalyse-Verfahren bieten Jain, Murty und Flynn [124] sowie im chemieinformatischen Kontext Downs und Barnard [125].

- *Unähnlichkeitsbasierte Methoden* beschreiben Diversität als eine Funktion der paarweisen Unähnlichkeit. Die Verfahren erlauben die Verwendung der meisten Deskriptoren, die für die Berechnung der paarweisen molekularen Ähnlichkeit genutzt werden. Im Gegensatz zu zell- und clusterbasierten Verfahren, die ähnliche Moleküle gruppieren, um eine Auswahl unähnlicher Substanzen treffen zu können, wird versucht, direkt eine Untermenge unähnlicher Substanzen auszuwählen. So wird die MaxiMin-Methode verwendet, um die minimale paarweise Distanz zwischen den Reagenzien zu maximieren [20].

Seit den Anfängen der kombinatorischen Chemie hat sich das Designziel allerdings grundlegend verändert. Zunächst sollten möglichst große und diverse Bibliotheken generiert werden. Die robotergestützte Synthese und Testung dieser Bibliotheken führte zwar zu einem sprunghaften Anstieg an synthetisierten Molekülen, aber nicht zu dem erwarteten Anstieg an bekannten aktiven Substanzen [126]. Wenn die synthetisierten

Moleküle jedoch keine biologische Aktivität aufweisen oder sich nicht zur Weiterentwicklung als Leitstruktur eignen, ist die Möglichkeit, Moleküle zu niedrigeren Kosten zu synthetisieren, nur von geringem Nutzen [127, 128].

Die Optimierung der Diversität einer Bibliothek erfolgt dementsprechend parallel zur Optimierung weiterer Kriterien. Insbesondere die Einhaltung eines physikochemischen Profils ist erwünscht (siehe Kapitel 2.3.1). Die Bewertung der Diversität geschieht dann beispielsweise durch die Berechnung der paarweisen Unähnlichkeit oder durch die Quantifizierung der Abdeckung eines partitionierten chemischen Raumes [10].

Gerade in der Phase der Leitstrukturidentifizierung ist Diversität zwar weiterhin ein wichtiges, jedoch nicht mehr das alleinige Auswahlkriterium [24, 25, 40].

2.4 Ausgewählte Verfahren aus der Literatur

Nachfolgend werden einige Verfahren zum Design kombinatorischer Bibliotheken kurz vorgestellt:

GALOPED [129] verwendet einen Genetischen Algorithmus für den Entwurf diverser Bibliotheken. Die Bibliotheken werden als Chromosomen kodiert. Um die Diversität zu bewerten, werden die Produkte einer Bibliothek enumeriert, mittels 2D-Deskriptoren geclustert und die Anzahl der entstehenden Cluster gezählt.

PLUMS [130] dient der Auswahl einer Subbibliothek, die sowohl möglichst effizient als auch effektiv ist. Zur Bewertung der Effektivität einer Subbibliothek wird die Anzahl der erwünschten Produkte durch die Anzahl der insgesamt als erwünscht definierten Substanzen geteilt. Hierfür werden zunächst die erwünschten Substanzen definiert und die zugehörigen Synthesebausteine in einer Startbibliothek zusammengefasst. Dies ist die effektivste Bibliothek. Für die Bewertung der Effizienz wird die Anzahl der erwünschten Produkte durch die Anzahl aller resultierenden Produkte geteilt. Um die Effizienz der Bibliothek zu erhöhen, werden iterativ Synthesebausteine entfernt.

PICCOLO [131] verwendet Simulierte Abkühlung und eine gewichtete Summe zur Bewertung der Kriterien. Betrachtet wird unter anderem die Ähnlichkeit zu Leitstrukturen, Reagenziendiversität, Produktneuheit, physikochemische Eigenschaften und Reagenzienpreis.

ULTRAFAST [12] verwendet einen Greedy-Algorithmus, der sequentiell für jede Substitutionsstelle die Auswahl der Reagenzien so lange optimiert, bis keine Verbesserung mehr möglich ist. Optimierungskriterien sind Ähnlichkeit und physikochemische Eigenschaften. Auch die Bindungsaffinität kann in Betracht gezogen werden.

MoSELECT [26] basiert auf SELECT [132] und verwendet einen Genetischen Algorithmus, um gleichzeitig unterschiedliche Kriterien wie Diversität, physikochemische Eigenschaften und die Syntheseschwierigkeit zu optimieren. Dabei wird eine paretooptimale Lösung gesucht. Zusätzlich können die Bibliotheksgröße und die Anzahl der Reagenzien an jeder Position als Optimierungskriterien verwendet werden [133].

LD1.0 [134] verwendet einen Genetischen Algorithmus (Bibliotheken als Chromosom), sowie 2D-Deskriptoren für strukturelle Diversität, physikochemische Eigenschaften und DOCK [135] zur Bewertung der Bindungsaffinität.

WEALD [136] tauscht die Reagenzien iterativ aus. Die Wahrscheinlichkeit für die Auswahl der einzelnen Reagenzien wird währenddessen kontinuierlich angepasst. Zur Bewertung der Kriterien wird ein Wünschbarkeitsindex [64, 65] verwendet (siehe Kapitel 5.6).

GLARE [69, 70] verwendet einen deterministischen Greedy-Algorithmus um wie PLUMS die größtmögliche Subbibliothek auszuwählen. Zur effizienten Bewertung verwenden Truchon und Bayly additive physikochemische Eigenschaften.

COLIBREE [137] verwendet eine Teilchenschwarmoptimierung (PSO). Hierbei dient das Verhalten von Vogel- beziehungsweise Fischschwärmen bei der Nahrungssuche als Vorbild. Die Ähnlichkeit beziehungsweise Unähnlichkeit zu Referenzmolekülen wird mit dem CATS Deskriptor [138] berechnet.

GARLig [139] verwendet einen Genetischen Algorithmus und bewertet die Produkte anhand der berechneten Bindungsaffinität (AutoDock [140, 141] oder GOLD [142, 143]).

MEGALib [62] verwendet ebenfalls einen Genetischen Algorithmus zur Suche nach dem Pareto-Optimum. Für die Berechnung der Ähnlichkeit zu Referenzmolekülen wird FUZZEE und für die Bewertung der Bindungaffinität wird GlamDock verwendet. Beides ist Teil der Chil2-Plattform [144].

FTrees-FS [31] erlaubt das ähnlichkeitsbasierte Cherry-Picking in Fragmenträumen mit dem Feature-Tree-Deskriptor. In Kapitel 3 wird darauf im Detail eingegangen.

3

Suche nach ähnlichen Molekülen in virtuellen Fragmenträumen

Für den Entwurf von virtuellen fokussierten Bibliotheken müssen die Syntheseprotokolle und Reaktanten in einem geeigneten Format vorliegen. Hierfür eignen sich Fragmenträume, die auch die effizienten Suche nach ähnlichen Molekülen mit dem Feature-Tree-Deskriptor erlauben [31]. Da kombinatorische Bibliotheken als Sonderfall von Fragmenträumen modelliert werden können [89, 145], ist der Feature-Tree-Deskriptor ebenfalls interessant für das produktbasierte Bibliotheksdesign. Das in Kapitel 5 vorgestellte Verfahren zum Design von fokussierten Bibliotheken verwendet deshalb Fragmenträume als Eingabeformat und den Feature-Tree-Deskriptor zum Ähnlichkeitsvergleich.

3.1 Chemische Fragmenträume

Die Untersuchung des gesamten chemischen Raumes ist aufgrund seiner Größe praktisch unmöglich [98, 146]. Die Gesamtzahl an möglichen Molekülen (*Chemical Space*) liegt Schätzungen zufolge im Bereich bis zu 10^{200} [145–149]. Allein die virtuelle Enumeration der Moleküle mit weniger als zwölf Schweratomen (C, N, O, F) ergibt unter Berücksichtigung von Valenzzuständen, Synthetisierbarkeit und Stabilität mehr als 26,4 Millionen Strukturen beziehungsweise 110,9 Millionen Stereoisomere [150]. Um große Mengen von Molekülen virtuell speichern und durchsuchen zu können, ist eine effiziente Beschreibung notwendig, ohne dabei die Synthetisierbarkeit der Moleküle außer Acht zu lassen. Hierfür eignen sich virtuelle chemischen Fragmenträume.

Ein gängiger Ansatz für die Generierung von Fragmenträume ist die Verwendung von retrosynthetischen Regeln, um Moleküle in chemische Bausteine (Fragmente) zu zerlegen. Die resultierenden Fragmenträume werden als Eingabe für virtuelle Methoden verwendet. Sie bestehen aus einer Menge von Fragmenten und Regeln. Die einzelnen Regeln spezifizieren, welche Fragmente verbunden werden können und wie die Verknüpfung zu erfolgen hat. Da die beinhalteten Substanzen nur implizit dargestellt werden, bietet ein Fragmentraum die Möglichkeit, einen großen chemischen Unterraum effizient zu kodieren und abzuspeichern. Durch die Regeln sind dabei Aspekte der Synthetisierbarkeit automatisch berücksichtigt. Fragmentbasierte Ansätze erlauben es, diesen Unterraum effizient zu durchsuchen. Sie gehören zu den sogenannten *De-novo*-Verfahren, die neue Moleküle aus kleineren chemischen Bausteinen erzeugen. Die Verfahren unterscheiden sich anhand der Art und Größe der verwendeten Bausteine, wie der chemische Raum durchsucht wird, und wie die Qualitität der generierten Moleküle bewertet wird [151, 152]. Zahlreiche Übersichtsartikel [146, 151–153] befassen sich mit dem *De-novo*-Design. Fragmentbasierte Ansätze existieren unter anderem für das Enumerieren von Molekülen unter physikochemischen Randbedingungen (FragEnum [68]), den Grundgerüstwechsel (ReCore [154]) und die ähnlichkeitsbasierte Suche (TOPAS [155], Flux [156, 157] und FTrees-FS [31]). Außerdem existieren Programme für die strukturbasierte (FlexNovo [34]), pharmakophorbasierte Suche (QSearch [158]) sowie das fokussierte Bibliotheksdesign (COLIBREE [137]). Zaliani et al. [159] haben einige der hier erwähnten Verfahren in einem Arbeitsablauf kombiniert. Hierbei wurden ReCore [154] und Colibri [89] verwendet, um die Fragmente zu generieren.

3.1.1 Erzeugung von Fragmenten unter Verwendung retrosynthetischer Regeln

Ein etabliertes Verfahren zur Erzeugung von Fragmenten ist die Zerlegung von Molekülen mittels retrosynthetischer Regeln (*Shredding*). Dabei werden zunächst die zu schneidenden Bindungen identifiziert und anschließend simultan gespalten. Die entstehenden offenen Valenzen werden mit sogenannten Linkatomen markiert. Diese sind mit einem entsprechenden Linktyp ausgestattet. Ein erstes automatisiertes Verfahren war RECAP [160], welches zur Generierung eines Eingaberaumes für TOPAS [155] und FTrees-FS [31] verwendet wurde. Um die Fragmente zu generieren, wurden die Moleküle des *World Drug Index (WDI)* [161] zerlegt (siehe auch Abbildung 3.1). Weitere von der jeweiligen

Anwendung motivierte Regelwerke wurden von Mauser und Stahl [162] und Degen et al. [163] (BRICS) publiziert.

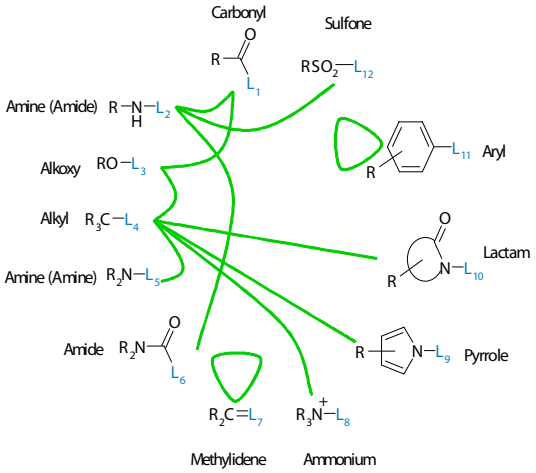


Abbildung 3.1: Fragment-Prototypen des FTrees-FS-Raumes [31] - Die Kompatibilität der Fragment-Prototypen wird durch die verbindenden Linien dargestellt. Bei den jeweiligen Prototypen sind unterschiedliche R-Gruppen möglich. Die R-Gruppen können wiederum weitere Linkatome beinhalten. Die Kreise stehen für Ringe unterschiedlicher Größe, sie können anneliert oder auch verbrückt sein. Die schematische Darstellung basiert auf der Anwendung der RECAP-Regeln [160]. Die Abbildung wurde von Rarey [164] adaptiert.

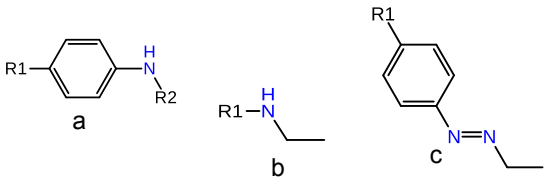


Abbildung 3.2: Verknüpfung von Fragmenten - Die Fragmente a (Anilin) und b (Ethylamin) werden zu Fragment c, einem Ethylazobenzol, verbunden. Die Linkatome werden verworfen und eine Doppelbindung entsteht. Um die neutrale Ladung der Atome aufrecht zu erhalten, wird jeweils ein Wasserstoff an den zu verbindenden Stickstoffen entfernt.

Fragmente unterscheiden sich von Molekülen somit nur aufgrund ihrer Linkatome. Jedes Linkatom besitzt einen Linktyp, welcher wiederum die Kompatibilität festlegt. Ergänzend kommt der Regelsatz hinzu, der steuert, ob zwei Fragmente, beziehungsweise Linktypen, kompatibel sind und wie sie verknüpft werden (siehe Abbildung 3.1). Zudem kann für jeden Linktyp eine terminale Gruppe definiert werden, die verwendet wird,

um das Fragment an dieser Stelle abzuschließen. Exemplarisch zeigt Abbildung 3.2 die Verknüpfung zweier Fragmente. In dem Beispiel kommt zu einer Änderung der Bindungsordnung der neu verbundenen Atome. Prinzipiell ist dies möglich, verdeutlicht aber, dass eine sorgfältige Modellierung des Raumes speziell für Programme notwendig ist, welche die Fragmente während der Berechnung nicht wirklich zusammensetzen, da sonst die Eigenschaften der Moleküle nicht richtig vorhergesagt werden können (siehe auch Kapitel 4.4). Zudem muss dem Programm ein chemisches Modell zugrunde liegen, welches falsche Valenzzustände erkennt und gegebenenfalls korrigiert.

3.1.2 Der Fragmentraum als Sammlung kombinatorischer Bibliotheken

Die Regeln eines Fragmentraumes beschreiben zwar theoretisch die synthetische Zugänglichkeit eines daraus generierten Moleküls. Dennoch scheitert die Durchführbarkeit der Synthese oftmals an einem oder mehreren der folgenden Probleme [145]:

- Die aktuelle Verfügbarkeit der Reaktionspartner ist nicht bekannt.
- Es kann Kombinationen von Fragmenten geben, die sich so nicht synthetisieren lassen, obwohl die Verknüpfungsregel auf einer bekannten Reaktion beruht. Da die Verknüpfungsregeln im Fragmentraum allgemein anwendbare Reaktionen berücksichtigen, können unter Umständen spezifische Reaktionen und Reaktionsbedingungen nicht wiedergegeben werden.
- Die Kombination von Fragmenten basierend auf unterschiedlichen Reaktionen kann dazu führen, dass das Produkt nicht synthetisierbar ist, da sich die Reaktionen gegenseitig ausschließen.

Basieren die Regeln des Fragmentraumes dagegen auf bekannten Syntheseprotokollen der kombinatorischen Chemie, kann die synthetische Zugänglichkeit mehr oder weniger garantiert werden. So wurden zwei Ansätze publiziert, die Fragmenträume verwenden, die eine Sammlung kombinatorischer Bibliotheken beinhalten, um darin mit dem Feature-Tree-Deskriptor nach ähnlichen Molekülen zu suchen [89, 145]. Diese virtuellen kombinatorischen Bibliotheken ergeben sich durch die Kombination verschiedener Reaktanten in einem einheitlichen Reaktionsschema. Die Reaktionen wiederum sind durch die Verknüpfungsregeln des Fragmentraumes kodiert. Die Syntheseprotokolle werden dadurch für die *In-silico*-Suche zugänglich.

3.2 Feature-Tree-Deskriptor

Bei dem Feature-Tree-Deskriptor [30, 165, 166] (*FTree*) handelt es sich um eine reduzierte Graphdarstellung (siehe Kapitel 2.2.5). Er repräsentiert den Molekülgraphen als einen ungerichteten, ungewurzelten Baum. Im Prinzip werden die durch drehbare Bindungen getrennten Baugruppen zu Knoten zusammengefasst. Die Knoten werden mit den physikochemischen und sterischen Eigenschaften der jeweiligen Substruktur annotiert. Die übergreifende Topologie und Konnektivität der Baugruppen bleibt dadurch erhalten, ohne die möglichen Konformationen betrachten zu müssen. Die Ähnlichkeit zwischen zwei Molekülen wird berechnet, indem die Knoten der Feature-Trees einander bestmöglich zugeordnet werden. Dies ist aufwändiger und somit zirka zwei bis drei Größenordnungen langsamer, als die Anwendung von Distanzmaßen und Ähnlichkeitskoeffizienten auf vektoriellen Repräsentationen [48].

Im Gegensatz zu Verfahren, die auf Substrukturen des Moleküls beruhen und Atome nur anhand ihres Elementes unterscheiden, sind zwei Feature-Tree-Knoten ähnlich, wenn die gleichen Interaktionen ausgebildet werden [22]. Ansonsten können physikochemische Äquivalenzen zwischen Atomtypen nicht erkannt werden [83, 167]. Durch die Abstraktion ist, wie zum Beispiel von Boehm et al. [89] gezeigt, der Austausch des Grundgerüstes möglich, so dass aktive Moleküle in anderen Strukturklassen gefunden werden können.

3.2.1 Generierung

Zur Generierung eines Feature-Tree werden zunächst die Atome auf die Knoten abgebildet. Dabei entspricht ein Knoten einer Menge von verbundenen Atomen, wobei alle Atome mindestens einem Knoten zugeordnet sind. Eine Kante zwischen zwei Knoten existiert, wenn die Knoten gemeinsame Atome besitzen oder zwei Atome benachbart sind. Einfach gebundene Atome – mit Ausnahme der Linkatome – werden dem Knoten ihres Bindungspartners zugeordnet. Dadurch sind Wasserstoffe niemals einem eigenen Knoten zugeordnet. Für die Zyklenfreiheit müssen die Ringsysteme des Moleküls detektiert und zerlegt werden. Da jedes Ringsystem genau einer Bizusammenhangskomponente entspricht, wird ein Zyklengraph generiert. Die Zyklen minimaler Länge werden mit einem Algorithmus ähnlich dem zur Bestimmung der Bizusammenhangskomponenten

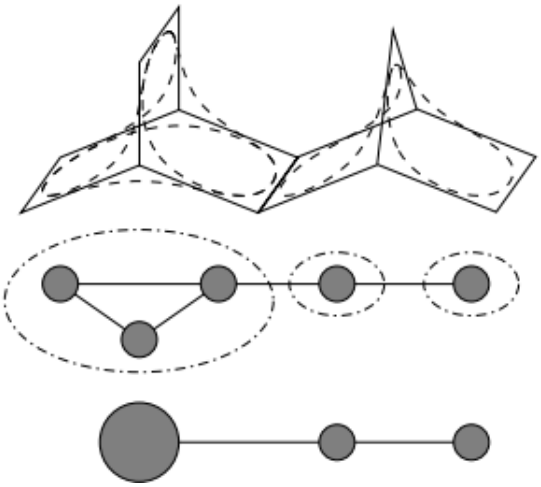


Abbildung 3.3: Zerlegung eines Ringsystems - Für das dargestellte Ringsystem werden zunächst die minimalen Zyklen (graue Linien) bestimmt. Für den resultierenden Zyklengraph werden wiederum die Bizusammenhangskomponenten bestimmt. Aus diesen wird der Feature-Tree generiert. Die Abbildung wurde adaptiert von Rarey und Dixon [30]

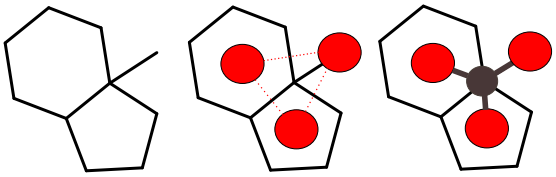


Abbildung 3.4: Beispiel für einen Leerknoten - Atome, die mehreren Ringen angehören, werden Brückenkopfatome genannt. In diesem Beispiel hat eines der beiden Brückenkopfatome eine azyklische Bindung. Um den bei der Generierung des Feature-Trees entstehenden Zyklus zu entfernen, wird ein Leerknoten eingefügt.

(siehe zum Beispiel Cormen et al. [168]) bestimmt. Diese bilden die Knoten des Zyklengraphens. Kanten werden zwischen zwei Knoten eingefügt, deren Zyklen mindestens ein gemeinsames Atom beinhalten. Enthält der Zyklengraph immer noch Zyklen, wird wiederum der Algorithmus verwendet, um diese Knoten miteinander zu verschmelzen. Abbildung 3.3 zeigt beispielhaft die Zerlegung eines Ringsystems. Unter Umständen ist es notwendig, leere Knoten einzufügen, um die Zyklenfreiheit garantieren zu können (siehe Abbildung 3.4).

Die Knoten eines Feature-Trees werden mit den physikochemischen und sterischen Eigenschaften der korrespondierenden Atome annotiert. Bei den verwendeten sterischen Eigenschaften handelt es sich um die Anzahl der Ringschlüsse und das approximierte Van-der-Waals Volumen. Zusätzlich wird die Anzahl der Atome gespeichert. Aber auch andere Eigenschaften wie die atomare Masse oder die Löslichkeit können theoretisch annotiert werden. Standardmäßig wird das Wechselwirkungssprofil von FlexX [117] genutzt (siehe Tabelle 3.1) Es beschreibt die Anzahl der möglichen Interaktionen, die ein

Interaktionstyp	Gewichtung
Wasserstoffbrückendonor	3
Wasserstoffbrückenakzeptor	3
Aromatisches Ringzentrum	1
Aromatisches Kohlenstoffatom, Methyl-Gruppe, Amid	1
Hydrophobes Atom	1

Tabelle 3.1: FlexX-Interaktionstypen [117] und ihre Gewichtung [30].

Molekülfragment mit einem Rezeptor eingehen kann. Diese Interaktionen werden in bestimmte Typen unterteilt. Zunächst werden die jeweiligen Wechselwirkungstypen identifiziert. Anschließend wird der Eintrag des jeweiligen Typs gesetzt. Hierfür wird Anzahl der Vorkommnisse wird mit einem Gewichtungsfaktor multipliziert, der die Wichtigkeit des Typs widerspiegelt. Für die Visualisierung wird den Knoten zudem eine räumliche Koordinate und ein Radius zugewiesen. Die Koordinate ist der Zentroid der korrespondierenden Atome. Die Größe eines Knotens wiederum ergibt sich aus der Anzahl der Schweratome der jeweiligen molekularen Baugruppe.

Abbildung 3.5 zeigt exemplarisch den Thrombin-Inhibitor MD-805 (PDB-Code [1] 1DWC) und den daraus generierten Feature-Tree, sowie die Knotenzuordnung des Ähnlichkeitsvergleiches mit dem Feature-Tree des Thrombin-Inhibitors NAPAP (PDB-Code 1DWD).

Der aus dem Molekülgraph generierte Feature-Tree beschreibt das Molekül auf der detailliertesten Auflösungsebene und ist für das Molekül eindeutig. Ebenso kann das Molekül durch einen Feature-Tree repräsentiert werden, der nur aus einem Knoten besteht. Dieser erlaubt eine schnelle Ähnlichkeitsabschätzung, da keinerlei topologische Informationen mehr enthalten sind. Ausgehend von dem aus dem Molekül generierten Feature-Tree, kann durch Kondensierung von Teilbäumen zu Knoten jedweder Detailgrad erreicht werden. Diese Eigenschaft wird von den im Folgenden beschriebenen Vergleichalgorithmen genutzt.

3.2.2 Paarweiser Ähnlichkeitsvergleich

Um die Ähnlichkeit zweier Feature-Trees zu berechnen, wird das maximal-gewichtete bipartite Matching (*maximum weighted bipartite matching*, siehe zum Beispiel Ottmann

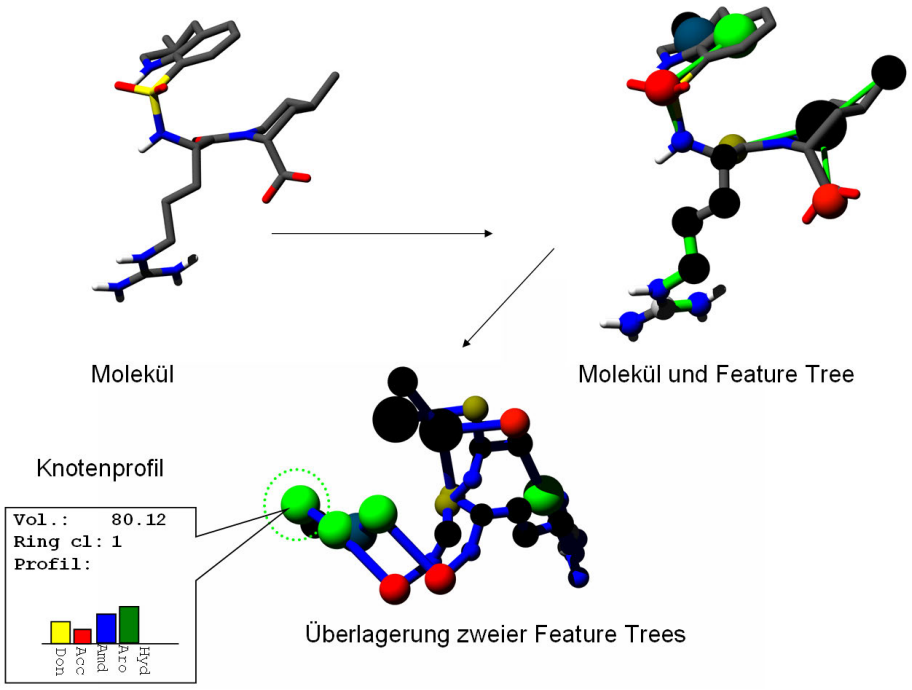


Abbildung 3.5: Feature-Tree-Generierung - Das Bild zeigt den Thrombin-Inhibitor MD-805 und den daraus entstehenden Feature-Tree, sowie die Überlagerung mit dem Feature-Tree eines weiteren Thrombin-Inhibitors (NAPAP). An den Knoten der Feature-Trees sind physikochemische und sterische Eigenschaften annotiert.

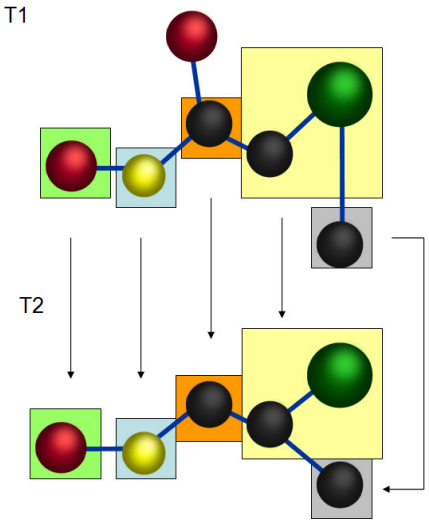


Abbildung 3.6: Topologie-erhaltendes Matching - Die Abbildung zeigt ein topologie-erhaltendes Matching der Feature-Trees T1 und T2. Die einzelnen Matches sind als farbige Kästen eingezeichnet und mit Pfeilen verbunden. Der obere rote Knoten von T1 ist Teil eines Nullmatches.

et al. [169] oder Cormen et al. [168]) bestimmt. Ein *Matching* $\mathcal{M}$ besteht aus einer Menge von einander zugeordneten Teilbäumen, den sogenannten *Matches*. Ziel ist es, die maximale Anzahl einander ähnlicher Teilbäume zu finden. Dabei können auch, insbesondere bei unterschiedlich großen Feature-Trees, partielle Matches entstehen, sogenannte *Nullmatches*. Es gilt die Einschränkung, dass ein Knoten jeweils nur maximal einem Match zugeordnet sein darf, damit ein Matching gültig ist. Zudem muss ein Matching topologie-erhaltend sein, so dass für zwei adjazente Knoten eines Baumes die im Matching zugeordneten Knoten ebenfalls adjazent sind. Werden die Matches zu einzelnen Knoten kollabiert, müssen die resultierenden Bäume isomorph sein. Ein Beispiel zeigt Abbildung 3.6. Das Matching von T1 und T2 besteht aus den farbig markierten Matches und ist topologie-erhaltend. Wäre bei T1 stattdessen der obere rote Knoten Teil des grünen Matches, ist das Matching nicht mehr topologie-erhaltend.

Ein Feature-Tree A sei im Folgenden definiert als eine zusammenhängende, zyklenfreie Menge von Knoten. Ein Teilbaum a sei definiert als eine zusammenhängende Teilmenge der Knoten von A. Um zwei Feature-Trees A, B zu vergleichen, wird die gewichtete Summe der Teilbaum-Matches des Matchings $\mathcal{M}$ berechnet. Als Gewichtung wird die Größe ($\text{size}(m)$) des jeweiligen Matches m verwendet. Der Ähnlichkeitswert wird normalisiert, indem durch die Gesamtgröße der Matches geteilt wird. Über den Gewichtungsfaktor $u \in [0, 1]$ kann zudem gesteuert werden, wie stark Nullmatches in die Bewertung einfließen.

$$S_M(A, B) = \frac{\sum\limits_{m \,\epsilon\, M} \text{size}(m) \cdot \text{sim}(m)}{u(\text{size}(A) + \text{size}(B)) + (1 - u) \sum\limits_{m \,\epsilon\, M} \text{size}(m)}$$

Die Ähnlichkeit zweier Teilbäume a, b eines Matches $m = (a, b) \in \mathcal{M}$ wird mit folgenden Formeln berechnet. Der Gewichtungsfaktor $s \in [0, 1]$ steuert den Einfluss der sterischen und physikochemischen Eigenschaften.

$$\text{sim}(a, b) = s(c_{\text{sterisch}}(a, b)) + (1 - s)(c_{\text{chemisch}}(a, b))$$

Die sterischen Eigenschaften werden mit der folgenden Formel 3.3 verglichen. Derzeit werden das Van-der-Waals-Volumen (Formel 3.1) und die Anzahl der Ringschlüsse (Formel 3.2) verwendet. Dabei berechnen die Funktionen $\text{vol}(a)$ das Van-der-Waals-Volumen

und $\mathrm{rc}(a)$ die Anzahl der Ringschlüsse des Teilbaumes a.

$$c_{\mathrm{vol}}(a,b) = \begin{cases} 1, & \text{wenn } a+b=0 \\ \frac{2\cdot\min(\mathrm{vol}(a),\mathrm{vol}(b))}{\mathrm{vol}(a)+\mathrm{vol}(b)}, & \text{sonst} \end{cases} \tag{3.1}$$

$$c_{\mathrm{rc}}(a,b) = \begin{cases} 1, & \text{wenn } a+b=0 \\ \frac{2\cdot\min(\mathrm{rc}(a),\mathrm{rc}(b)+1)}{\mathrm{rc}(a)+\mathrm{rc}(b)+2}, & \text{sonst} \end{cases} \tag{3.2}$$

$$c_{\mathrm{sterisch}}(a,b) = c_{\mathrm{vol}}(a,b)c_{\mathrm{rc}}(a,b) \tag{3.3}$$

Die Ähnlichkeit zweier chemischer Profile wird mit der folgenden Formel berechnet:

$$c_{\mathrm{chemisch}}(a,b) = \begin{cases} 1, & \text{wenn } \sum_i (a_i+b_i)=0 \\ 2\frac{\sum_i \min(a_i,b_i)}{\sum_i (a_i+b_i)}, & \text{sonst} \end{cases} \tag{3.4}$$

Für die Berechnung eines Matchings $\mathcal{M}$ stehen die drei Algorithmen **Split-Search** [30], **Match-Search** [30] und **Dynamic-Match-Search** [170, 171] zur Verfügung. Im Kontext dieser Arbeit wird näher auf den Match-Search-Algorithmus eingegangen, da er zur Ähnlichkeitssuche in Fragmenträumen verwendet werden kann [31].

3.2.3 Match-Search-Algorithmus

Die Eingabe des rekursiven Match-Search-Algorithmus sind zwei gewurzelte Teilbäume. Der Algorithmus teilt die gewurzelten Bäume in Blöcke einer bestimmten Größe und stoppt, sobald keine Teilbäume mehr zugeordnet werden können.

Zunächst wird ein initialer Schnitt bei beiden Feature-Trees durchgeführt. Die ungerichteten Kanten des Feature-Trees werden intern durch ein Paar antiparalleler, gerichteter Kanten repräsentiert, so dass durch einen gerichteten Schnitt jeweils zwei gerichtete, disjunkte Teilbäume entstehen. Ein *Split* besteht aus zwei gerichteten Schnitten, einem in jedem Feature-Tree. Partiell ist der Split, wenn einer der Feature-Trees einem der beiden Teilbäume des anderen vollständig zugeordnet wird. Um die optimale Zuordnung zu erhalten, müssen theoretisch alle möglichen Splits prozessiert werden. Bei einem Paar von Feature-Trees mit n beziehungsweise m Knoten ergeben sich $4(n-1)(m-1)$ mögliche Eingaben für den rekursiven Aufruf. Der Algorithmus beschränkt jedoch den Suchraum, da nur Splits erlaubt sind, bei denen die resultierenden Teilbäume in einem bestimmten Größenverhältnis zueinander stehen. Die Größe eines Teilbaumes ergibt sich aus der Menge der Schweratome, die er repräsentiert. Beginnend mit einem möglichst balancierten Split wird eine ungerichtete Kante herausgenommen und die adjazenten

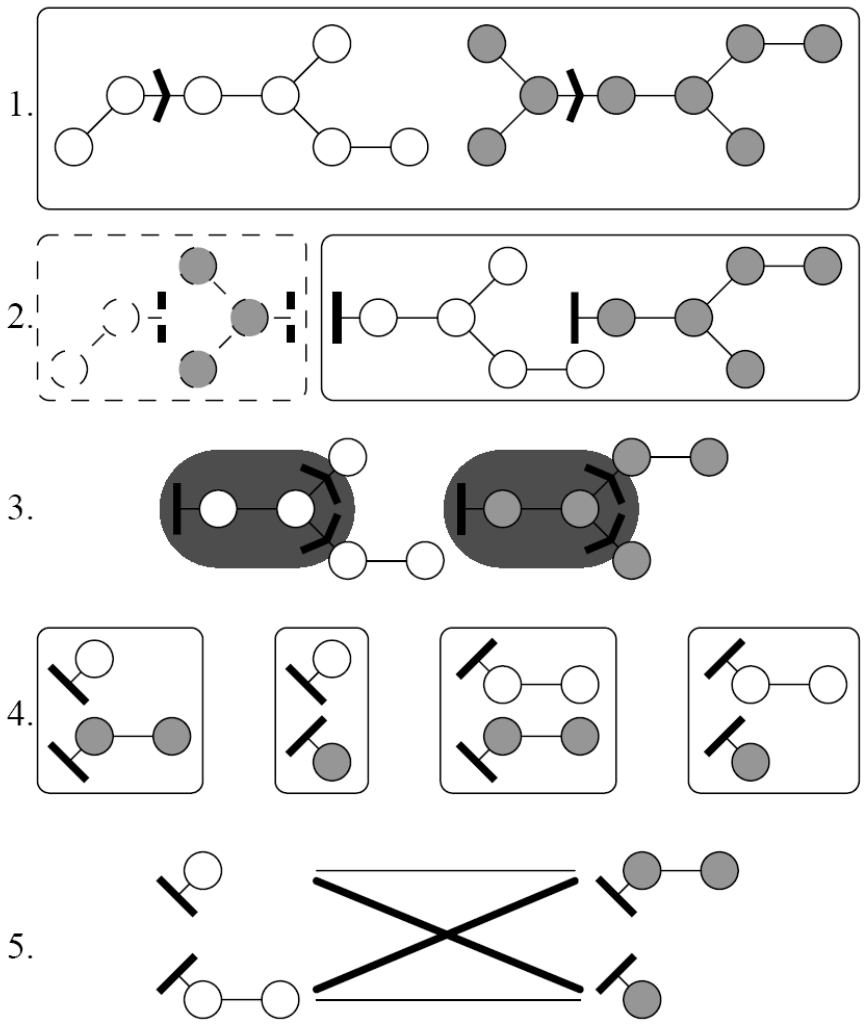


Abbildung 3.7: Match-Search-Algorithmus - Der Match-Search-Algorithmus teilt sich grundlegend in die folgenden fünf Phasen auf: (1) Die initialen Splits werden bestimmt. Die Abbildung zeigt einen möglichen Split. (2) Erster rekursiver Aufruf für die jeweiligen gewurzelten Teilbaumkombinationen. Im Folgenden wird die rechte Kombination weiter betrachtet. (3) Das Matching wird um ein Match erweitert, welches beide Wurzelknoten enthält. Dadurch sind die Matches zusammenhängend. (4) Rekursiver Aufruf für alle gewurzelten Teilbaumkombinationen, die durch die Separation des Matches entstehen. (5) Auswahl der Teilbaumkombination, die den höchsten Ähnlichkeitswert besitzt (hier dargestellt durch die dicker gezeichneten Linien). Die Abbildung wurde übernommen von Rarey und Dixon [30].

Knoten als Wurzelknoten markiert (siehe Abbildung 3.7, Schritt 1). Anschließend wird der rekursive Teil des Algorithmus mit zwei Wurzelknoten aufgerufen (siehe Abbildung 3.7, Schritt 2). Um die beiden Teilbäume einander zuzuordnen, wird zunächst versucht das Match der beiden Wurzelknoten um adjazente Knoten zu erweitern. An den Grenzen dieses Erweiterungs-Matches erfolgen erneut Schnitte, um das Match zu separieren (siehe Abbildung 3.7, Schritt 3). Der Algorithmus wird anschließend für jede mögliche Kombination der gewurzelten Teilbäume aufgerufen, die dabei entstanden sind, um die nächsten Erweiterungsmatches zu finden (Abbildung 3.7, Schritt 4). Können keine Teilbäume mehr zugeordnet werden, stoppt die Rekursion. Das resultierende Matching ist zusammenhängend. Interne Nullmatches sind nicht möglich, da die Wurzelknoten immer in den Matches enthalten sind.

Die Ähnlichkeit zweier Teilbäume ergibt sich folglich aus der Ähnlichkeit des Matches und der Kombination der darunterliegenden Teilbäume, die den höchsten Ähnlichkeitswert besitzt (Abbildung 3.7, Schritt 5). Der maximale Ähnlichkeitswert, der durch Setzen eines der initialen Splits berechnet wurde, beschreibt die Ähnlichkeit der beiden Feature-Trees. Die Ausgabe ist zum einen ein Ähnlichkeitswert im Intervall [0,1] und zum anderen eine Liste von Schnitten und Matches, welche die Unterteilung und Zuordnung der Teilbäume beschreiben. Abbildung 3.7 skizziert die einzelnen Schritte des Algorithmus, der Pseudocode ist in Algorithmus 3.1 aufgeführt.

Die Eingabe des Algorithmus entscheidet sich immer nur durch den Schnitt, der die Teilbäume separiert. Die optimale Zuordnung zweier Teilbäume wiederum beruht auf der optimalen Zuordnung der nächst kleineren Teilbäume, so dass Memoisation (siehe beispielsweise Cormen et al. [168]) und dynamische Programmierung [172] genutzt werden können. Der Match-Search ist ein rekursiver Algorithmus mit Memoisation. Bereits berechnete Teilergebnisse werden in einer Matrix zwischengespeichert, so dass die Ähnlichkeit jeder Teilbaumkombination nur einmal berechnet werden muss (siehe Abbildung 3.8). Dies erlaubt auch die Indexierung von Teilbäumen in großen Datensätzen [166] und die Suche in Fragmenträumen [31].

3.2.4 Ähnlichkeitssuche in Fragmenträumen

Der Match-Search-Algorithmus wurde bereits erfolgreich zur Ähnlichkeitssuche in Fragmenträumen [31], sowie Fragmenträumen bestehend aus kombinatorischen Bibliotheken

Aufruf : `match_search_similarity`(t_1, t_2)
Eingabe : Feature Trees f_1 und f_2
Ausgabe : Maximaler Ähnlichkeitswert zwischen f_1 und f_2

Initialisiere Matrix mat
$S \leftarrow$ `find_splits`(f_1, f_2)
foreach $s \in \mathcal{S}$ **do**
    $(f_{11}, f_{22})(f_{12}, f_{21}) \leftarrow$ wende s an
    `sv`$(s) \leftarrow$`recursive_match_search`$(f_{11},\ f_{22},\ mat)\ \oplus$ `recursive_match_search`$(f_{12},\ f_{21},\ mat)$
return $max\{$`sv`$(s) \mid s \in S\}$

Aufruf : `recursive_match_search`$(t_1,\ t_2,\ mat)$
Eingabe : Teilbäume t_1 und t_2, Matrix mat
Ausgabe : Maximaler Ähnlichkeitswert zwischen t_1 und t_2

if $mat[t_1, t_2]$ **then**
    return `sv`$(mat[t_1, t_2])$
$\mathcal{M} \leftarrow$ `find_matches`(t_1, t_2)
foreach $m \in \mathcal{M}$ **do**
    Schneide alle Kanten, die notwendig sind, um m zu separieren
    $\mathcal{R}_i \leftarrow$ Teilbäume von t_i, die von m separiert sind
    foreach $r_1 \in \mathcal{R}_1 \wedge r_2 \in \mathcal{R}_2$ **do**
        `sv`$(mat[r_1, r_2]) \leftarrow$ `recursive_match_search`$(r_1,\ r_2,\ mat)$
    $\mathcal{W} \leftarrow$ maximal gewichtetes bipartites Matching zwischen $\mathcal{R}_1$ und $\mathcal{R}_2$. wobei die jeweiligen Ähnlichkeitswerte `sv`$(mat[r_1, r_2])$ als Gewichte dienen, welche wiederum mit $\oplus$ addiert werden.
    $\mathcal{U}_i \leftarrow$ Teilbäume von s, die nicht in $\mathcal{W}$ zugeordnet wurden
    `sv`$(m) \leftarrow$ `sim`$(m) \oplus (\oplus_{r \in \mathcal{U}_1 \cup \mathcal{U}_2}$ `sim`$(r, NULL)) \oplus (\oplus_{r \in \mathcal{R}_1 \setminus \mathcal{U}_1}$`sv`$(mat[r, \mathcal{W}(r)]))$
`sv`$(mat[t_1, t_2]) \leftarrow\ max\{$`sv`$(m) \mid m \in \mathcal{M}\}$
return `sv`$(mat[t_1, t_2])$

Algorithmus 3.1 : Pseudocode des Match-Search-Algorithmus. Die Funktion `match_search_similarity` berechnet die Ähnlichkeit zweier Feature-Trees. Zunächst werden mit der Funktion `find_splits` initiale, nicht partielle Splits gesetzt. Für jeden dieser Schnitte wird `recursive_match_search` gestartet, um die Ähnlichkeit rekursiv zu berechnen. Als Eingabe dienen die gewurzelten Teilbäume t_1 und t_2. Intern werden die Funktionen `find_matches`, `sim` und $\oplus$ aufgerufen. Die mit `find_matches` generierten Matches enthalten die Wurzelknoten sowie mögliche Erweiterungsmatches. `sim` berechnet den direkten Ähnlichkeitswert und $\oplus$ kombiniert die Ähnlichkeitswerte verschiedener Matches. Des Weiteren steht `sv` für den Zugriff auf den jeweiligen Ähnlichkeitswert. Zur Vereinfachung wird die Speicherung von Splits und Matches in der Matrix nicht gezeigt. Der Pseudocode ist angelehnt an Rarey und Dixon [30].

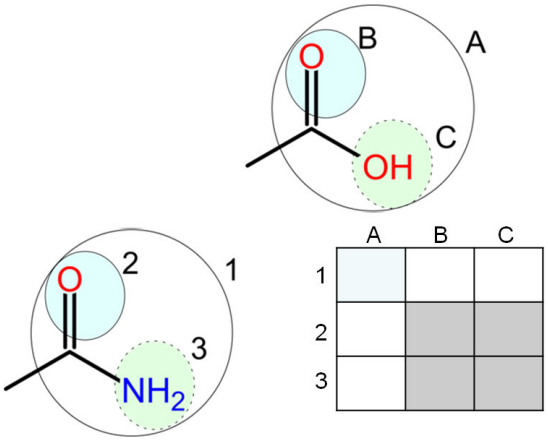


Abbildung 3.8: Abhängigkeit der Vergleiche - Unter der Bedingung, dass die Wurzelknoten aufeinander gelegt werden, basiert die optimale Zuordnung der Knoten der Teilbäume A und 1 auf der optimalen Zuordnung der nächstkleineren Teilbäume B, C und 2, 3. Dafür muss zunächst die Ähnlichkeit der Kombinationen B-2 und C-3 sowie B-3 und C-2 berechnet werden. Diese Werte werden in der Matrix gespeichert und wiederverwendet.

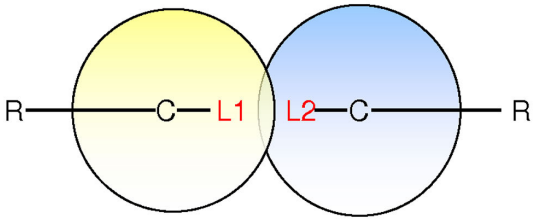


Abbildung 3.9: Unterschiede bei der Berechnung der Van-der-Waals-Kugeln - Beim Vergleich auf Fragment-Ebene fließt der Überlapp der Van-der-Waals-Kugeln der zum Linkatom adjazenten Atome doppelt ein, da der Bindungspartner zum Zeitpunkt der FTree-Generierung nicht bekannt ist. Die berechneten Ähnlichkeitswerten können dadurch von den Werten beim Vergleich zweier Moleküle abweichen.

[89, 145], angewandt. Der Vorteil liegt vor allem darin, dass die Moleküle für den Ähnlichkeitsvergleich nicht explizit aus den Fragmenten generiert werden müssen. Stattdessen kann die Ähnlichkeit des resultierenden Moleküls zur Anfrage direkt aus der Zuordnung der Fragmente abgeleitet werden. Dafür werden die Linkatome durch einen entsprechenden Linkknoten im jeweiligen Feature-Tree repräsentiert. Nachfolgend wird die Ähnlichkeitssuche in Fragmenträumen beschrieben.

Für den paarweisen Ähnlichkeitsvergleich zwischen einem Anfragemolekül und einem Produkt, welches sich aus mehreren Fragmenten zusammensetzt, wird zunächst das Basisfragment bestimmt. Dann wird die Ähnlichkeit zwischen den Fragment-FTrees, die mit den Linkknoten des Basisfragments verknüpft werden sollen, und allen möglichen Teilbäumen des Anfrage-FTrees berechnet und in die Matrix eingetragen. Als Eingabe des rekursiven Match-Search-Algorithmus dienen die aus dem Linkknoten dieses Fragment-FTree ausgehende Kante und die Wurzelkante des jeweiligen Anfrage-Teilbaums. Erst dann wird der Vergleich, wie in Abschnitt 3.2.3 beschrieben, durchgeführt. Hat der verwendete Fragment-FTree weitere Linkknoten, muss die Matrix dieses Vergleiches ebenfalls rekursiv vorprozessiert werden. Für eine effiziente Anfrage im Suchraum speichert FTrees-FS die berechneten Werte in einem Zwischenspeicher. Um Rekursionen zu vermeiden, werden die Teilbäume des Anfragemoleküls entsprechend ihrer Größe betrachtet. Ein Vergleich wird nur durchgeführt, wenn die beiden Teilbäume in einem bestimmten Größenverhältnis zueinander liegen. Um k ähnliche Moleküle zu generieren, werden aus dem Zwischenspeicher die Fragmente gewählt, die am ähnlichsten zu der jeweiligen Teilstruktur der Anfrage sind. Das passiert immer dann, wenn der rekursive Match-Search einen Linkknoten prozessiert. Zu beachten ist, dass mit dieser heuristischen Methode zwar das ähnlichste Molekül gefunden werden kann, jedoch nicht unbedingt die k ähnlichsten.

Des Weiteren können sich die berechneten Ähnlichkeitswerte von denen unterscheiden, die der Vergleich der jeweiligen Moleküle zurückliefert. Zum einen kann bei der Generierung der Fragment-FTrees das Van-der-Waals-Volumen nur approximiert werden (siehe Abbildung 3.9). Zum anderen werden nicht alle möglichen initialen Splits beim Vergleich zweier Moleküle gesetzt und somit der Suchraum beschränkt [30]. Bei der Fragmentraumsuche werden dagegen die Schnitte nur im initialen Fragment gesetzt.

4

Entwicklung eines neuen Chemie-Modells

Moleküle und Fragmente können aus unterschiedlichen Dateiformaten eingelesen werden. Die gängigsten Dateiformate für Moleküle sind Daylight SMILES [173], Symyx SDF [174] und Tripos MOL2 [175]. Sie basieren auf verschiedenen chemischen Modellen, so dass das gleiche Molekül auf unterschiedliche Art und Weise repräsentiert wird. Um die Moleküle auf eine konsistente und vom Format unabhängige interne Repräsentation zu bringen, ist ein robustes chemisches Modell notwendig. Aus diesem Grund wurde die *NAOMI*-Bibliothek [35] und der darauf basierende Dateiformatkonverter entwickelt[1]. Zunächst soll auf das bestehende Modell der Flex*-Bibliothek eingegangen werden.

4.1 Modell der Flex*-Bibliothek

In der Flex*-Bibliothek werden Moleküle durch Graphen repräsentiert. Die Atome werden durch Knoten und die kovalenten Bindungen durch ungerichtete Kanten dargestellt. Das atombasierte Chemie-Modell beruht auf der Erkennung der relevanten chemischen Substrukturen und der Zuordnung spezieller Atomtypen, den sogenannten Sybyltypen des MOL2-Formats [175]. Die relevanten chemischen Substrukturen werden anhand von SMARTS-Mustern [72] definiert[2]. Die SMARTS-Sprache bietet dem Nutzer die Möglichkeit mit dem Programm zu interagieren und beispielsweise bekannte toxische Moleküle mit Hilfe von SMARTS-Mustern aus einem Datensatz herauszufiltern. Bei Zugriff auf

[1] In Anhang C.1 wird genauer auf die Implementierung und beteiligte Personen eingegangen.

[2] Für die Substruktursuche kann zum Beispiel der Ullmann- [176] oder der VF2-Algorithmus [177] verwendet werden.

die SMARTS-Muster kann der Nutzer zudem das Chemie-Modell der Flex*-Bibliothek grundlegend beeinflussen.

Da ein Molekül sowohl in lokalisierter als auch in delokalisierter Repräsentation vorliegen kann, werden Transformationsregeln angewendet. Ein Molekül kann so von der einen in die andere Repräsentation überführt werden. Die Kodierung dieser Regeln erfolgt in der Flex*-Bibliothek ebenfalls mit SMARTS-Ausdrücken. Prinzipiell muss dafür in beide Richtungen ein vollständiges Regelwerk vorliegen. Dabei darf eine mehrfache Konvertierung das Molekül nicht verändern. Dies zu überprüfen und herauszufinden, welche SMARTS-Ausdrücke überlappen oder ungewollt weitere Substrukturen treffen, ist aufgrund der Vielzahl sehr unterschiedlicher Moleküle, der hohen Zahl eingesetzter SMARTS-Ausdrücke und der darin enthaltenen Platzhaltersymbole nicht möglich.

Auch zur Generierung einiger Deskriptoren wie dem TPSA [114] und dem aLogP [178, 179] oder den Knotenprofilen des Feature-Tree-Deskriptors werden jeweils eigene SMARTS-Regelsätze definiert. Die unterschiedliche Zuordnung von Eigenschaften zu Substrukturen in Abhängigkeit vom verwendeten Programm oder Algorithmus führt zu Inkonsistenzen in der chemischen Modellierung. Ein Beispiel ist die Aromatizität. Während der Molekülinitialisierung wird ein Algorithmus zur Erkennung aromatischer Substrukturen verwendet und die zugehörigen Atome als aromatisch markiert. Bei der Generierung von Feature-Trees wird jedoch auf SMARTS-Ausdrücke zurückgegriffen, so dass unter Umständen andere Atome als aromatisch markiert werden.

Zudem bietet die Flex*-Bibliothek dem Anwender eine mehrstufige Initialisierung, bei der einzelne Stufen wie die Delokalisierung oder Neuberechnung der Sybyltypen, individuell an- beziehungsweise abgestellt werden können. Ein Molekül kann dadurch in unterschiedlichen Zuständen vorliegen. Aus diesem Grund sollten bei einer Ähnlichkeitssuche für die Anfrage- und Datenbankmoleküle die gleichen Initialisierungsschritte erfolgen. Werden unterschiedliche Eingabeformate verwendet, sind bestimmte Initialisierungschritte erforderlich, um die Repräsentationen anzugleichen. Ein Benzolring wird zum Beispiel im SD-Format mit alternierenden Doppelbindungen beschrieben. Im MOL2-Format können die Bindungen auch explizit als aromatische Bindungen markiert werden. Dies macht die Verwendung einer abgestufte Initialisierung schwierig.

Anhand der folgenden drei Minimalbeispiele werden die Limitierungen des Flex*-Modells deutlich:

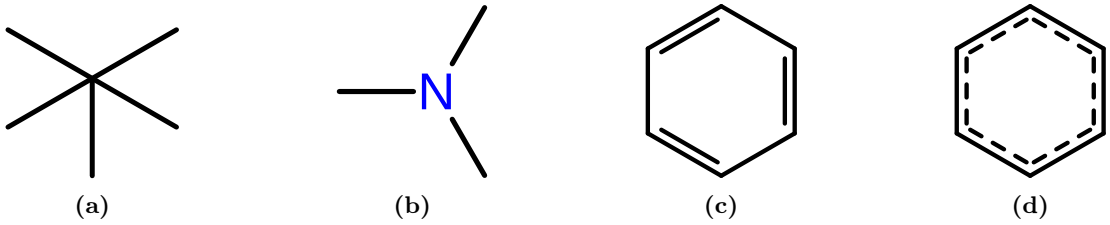


Abbildung 4.1: Probleme und Inkonsistenzen - (a) Falsche Valenzzustände, wie zum Beispiel ein fünfbindiger Kohlenstoff, werden nicht abgelehnt. (b) Stickstoffe mit drei Einfachbindungen werden nicht als Akzeptoren erkannt. Für den Benzolring ergeben sich in lokalisierter (c) beziehungsweise delokalisierter Darstellung (d) unterschiedliche Knoten-Volumina.

- Atome mit falschem Valenzzustand werden nicht abgewiesen (Abbildung 4.1a). Problematisch ist dies insbesondere bei Linkatomen, da diese endständig sein müssen.

- Stickstoffe mit drei Einfachbindungen werden bei der FTree-Generierung nicht als Wasserstoffbrückenakzeptoren erkannt (Abbildung 4.1b). Durch Abgleich mit dem *NAOMI*-Modell zeigte sich, dass das betreffende SMARTS-Muster fehlt.

- Das berechnete FTree-Knotenvolumen (94,33 beziehungsweise 94,76) unterscheidet sich für den Benzolring, je nachdem ob die lokalisierte (Abbildung 4.1c, SMILES: "C1=CC=CC=1") oder die delokalisierte (Abbildung 4.1d, SMILES: "c1ccccc1") Darstellung des Moleküls gelesen wurde und dementsprechend der Sybyltyp $C.am$ beziehungsweise $C.2$ zugewiesen wurde.

4.2 Anforderungen an das *NAOMI*-Modell

Zur konsistenten Beschreibung von Molekülen aus Datei-Formaten müssen die eingelesenen Moleküle auf eine eindeutige Repräsentation gebracht werden. Insbesondere müssen die funktionellen Gruppen in unterschiedlichen Darstellungen erkannt, unter Umständen korrigiert und auf eine universelle Beschreibung zurückgeführt werden.

Unter Berücksichtigung der in dieser Arbeit entwickelten Anwendung waren deshalb bei der Entwicklung von *NAOMI* die folgenden Kriterien wichtig:

- Chemische Validierung und Korrektur

- Konsistente und unabhängige Repräsentation der Moleküle
- Konsistente physikochemische Eigenschaften und Deskriptoren
- Reihenfolgeunabhängigkeit der Eingabedaten

Aus softwaretechnischer Sicht und im Hinblick auf große Eingabemengen wie zum Beispiel bei der Verwendung von Fragmenträumen (siehe Kapitel 3.1) wurden zusätzlich die folgenden Designkriterien und nichtfunktionalen Anforderungen beachtet:

- Nebenläufige Verwendbarkeit
- Geringer Speicherbedarf der Strukturen
- Modularisierung
- Objektorientiertes Design
- Softwaretests

4.3 *NAOMI*-Modell

Auch bei *NAOMI* werden die Moleküle durch einen Graphen repräsentiert. Das atombasierte Chemie-Modell beschreibt ein Atom auf drei Ebenen:

- **Element:** Die Elementebene stellt Eigenschaften zur Verfügung, die ausschließlich auf dem chemischen Element des Atoms beruhen. Beispiele hierfür sind die Atommasse und die Anzahl der Valenzelektronen.
- **Valenzzustand:** Ein Valenzzustand beschreibt das Atom in einer Valenzstruktur. Er stellt ein valides Bindungsmuster des Atoms dar. Valenzzustände enthalten topologische Informationen wie die Art und Anzahl der Bindungen, die Anzahl der Wasserstoffe und die Formalladung.
- **Atomtyp:** Der Atomtyp erweitert den Valenzzustand, so dass Aromatizität sowie äquivalente Resonanzformen modelliert werden können.

Während der Initialisierung werden jedem Atom Element, Valenzzustand und Atomtyp zugewiesen. Außerdem werden funktionelle Gruppen, Ringe und Ringsysteme sowie mögliche Stereodeskriptoren identifiziert und annotiert. Unter Verwendung von Valenzzustand und Atomtyp ist es möglich, Moleküle nach außen sowohl in lokalisierter als auch delokalisierter Form zu repräsentieren. Eine Transformation ist nicht nötig.

Die konsistente, unabhängige Repräsentation der Moleküle wird durch sogenannte *Round-Trip-Tests* sichergestellt. Moleküle werden aus unterschiedlichen Dateiformaten eingelesen und wieder herausgeschrieben. Die neu generierten Dateien werden dann wiederum eingelesen und der *Unique SMILES* [173], eine kanonische Zeichenkettenrepräsentation[1], wird verglichen.

Für konsistente Deskriptoren ist es zudem notwendig, dass die Initialisierung und Annotation unabhängig von der Reihenfolge der Eingabedaten ist. Auf die Notwendigkeit konsistenter Deskriptoren wird in Kapitel 4.4 noch einmal genauer eingegangen.

Unter Verwendung heutiger Multikern-Architekturen sollte es möglich sein, dass Moleküle nebenläufig präprozessiert, initialisiert und verwendet werden können. Dafür müssen die Module und Methoden der Bibliothek eintrittsinvariant sein. Ein geringerer Speicherbedarf kann im Vergleich mit der bisherigen Molekülrepräsentation vor allem dadurch erreicht werden, dass atomspezifische Daten, wie zum Beispiel die Atommasse, nicht mehr direkt am Atom gespeichert werden. Stattdessen erfolgt der Zugriff über Element, Valenzzustand und Atomtyp, welche als statische Strukturen angelegt sind.

Eine strikte Modularisierung führt zu besserer Wiederverwendbarkeit, Übersichtlichkeit und geringerer Fehleranfälligkeit. Die Modultests erlauben die systematische Fehlersuche und Fixierung der Schnittstellen anhand positiver und negativer Testfälle. Werden Teile des Codes modifiziert, können so unerwünschte Nebeneffekte erkannt und das erwartete Verhalten spezifiziert werden. Anhang C beschreibt den grundlegenden Aufbau der in C++ entwickelten *NAOMI*-Bibliothek. Auch auf die Modularisierung der Bibliothek und das verwendete Testsystem wird eingegangen.

[1] Anzumerken ist, dass es theoretisch nicht-isomorphe Molekülgraphen gibt, die in die gleiche Zeichenkette resultieren [180].

4.4 Generierung konsistenter Deskriptoren

Bei der parallelen Verwendung mehrerer physikochemischer Eigenschaften und Deskriptoren, wie zum Beispiel beim Filtern von Molekülen (Kapitel 5.11) oder bei der multikriteriellen Optimierung (Kapitel 5.6), ist es von entscheidender Bedeutung, dass die verwendeten Deskriptoren auf dem gleichen chemischen Modell beruhen. Um konsistente Deskriptoren zu erhalten, werden sie unter Verwendung des Chemie-Modells generiert. Ein positiver Nebeneffekt ist, dass diese Deskriptoren bedeutend schneller generiert werden können. Da die benötigten Informationen direkt von den Valenzzuständen und Atomtypen abgelesen werden können, muss kein rechenintensives Subgraph-Matching durchgeführt werden. Die Berechnung des TPSA-Deskriptors [114] erfolgt zum Beispiel unter Betrachtung von Valenzzuständen, Aromatizität und der Ringgrößen. Für die Referenzmoleküle, die der TPSA-Publikation [114] als SMILES beigefügt sind, wurden die jeweiligen TPSA-Werte in einem Test fixiert.

4.4.1 Veränderte Generierung des Feature-Tree-Deskriptors

Das Hauptoptimierungskriterium bei dem in dieser Arbeit vorgestellten Verfahren (*LoFT*) ist die Ähnlichkeit zu anderen Molekülen. Somit muss insbesondere bei der Generierung der Feature-Tree-Deskriptoren auf Konsistenz zu den weiteren verwendeten Deskriptoren geachtet werden. Um dies zu erreichen, müssen die für die Generierung relevanten chemischen Eigenschaften aus dem gleichen chemischen Modell abgeleitet werden.

Deshalb wurde im Rahmen der vorliegenden Arbeit ein neues Modul zur Generierung von Feature-Trees entwickelt. Bei der Generierung werden die FlexX-Interaktionstypen [117] der jeweiligen Substruktur anhand des *NAOMI*-Chemie-Modells am jeweiligen Knoten annotiert. Wie von Rarey und Dixon [30] beschrieben, wird die Anzahl der möglichen Interaktionen für Wasserstoffbrückenakzeptoren und -donoren anhand der freien Elektronenpaare beziehungsweise der Wasserstoffatome berechnet. Zur Bestimmung der freien Elektronenpaare wird auf die Idealgeometrie (VSEPR-Modell [181]) des jeweiligen *NAOMI*-Atomtyps zurückgegriffen. Bei mesomeren Strukturen kann dadurch die Anzahl der freien Elektronenpaare für die Atome unabhängig von der jeweiligen lokalisierten Struktur bestimmt werden. So ergeben sich zum Beispiel bei der Carboxylatgruppe für die beide Sauerstoffatomen jeweils zwei freie Elektronenpaare. Zusätzlich werden Ringeigenschaften wie die Aromatizität nicht über die Atome annotiert, sondern über

die Ringe, die dem Knoten zugeordnet sind. Besonders die große Anzahl an möglichen Heterozyklen [182, 183] erschwert die Identifikation auf Basis von SMARTS-Mustern. Der Nachteil der internen Beschreibung ist, dass der Nutzer keinen direkten Zugriff auf das Modell hat. Prinzipiell ist es möglich, die *NAOMI*-Bibliothek so zu erweitern, dass Deskriptoren über SMARTS-Ausdrücke definiert werden. In diesem Fall können die SMARTS-Muster mit den internen Daten des Chemiemodells abgeglichen werden.

4.5 Implikationen für die Verwendung von Fragmenträumen

Um die Fragmente auf chemische Validität überprüfen zu können, wurde im Chemie-Modell der Elementtyp "Linker" eingeführt. Ein Linker darf stets nur über eine Einfach-, Doppel- oder Dreifachbindung verfügen. Dadurch werden beispielsweise Fragmente, bei denen ein zusätzlicher Wasserstoff an das Linkatom gebunden ist, abgelehnt. Während der Initialisierung muss das Atom formatspezifisch als Linkatom erkannt und der Elementtyp "Linker" zugewiesen werden. Viele der Algorithmen für Moleküle lassen sich dadurch auf Fragmente anwenden. So können die Linknamen in 2D-Abbildungen angezeigt werden, ohne dass der Fragmentraum beziehungsweise seine Regeln bekannt sein müssen.

Ein weiterer Vorteil des Chemie-Modells ist, dass die Informationen, welche die topologische und geometrische Modifikation durch die jeweilige Verknüpfungsregel beschreiben, intern im Modell abgebildet und somit als externe Eingabe obsolet sind. Exemplarisch sind die Anzahl der Wasserstoffe und der Hybridisierungszustand der neu verbundenen Atome zu nennen.

Zusammengefasst bietet das *NAOMI*-Modell die Möglichkeit, Fragmente konsistent und korrekt zu verknüpfen.

5

Konzepte und Methoden für den Entwurf fokussierter Bibliotheken

Im folgenden Kapitel werden die algorithmischen Konzepte für den Entwurf fokussierter kombinatorischer Bibliotheken vorgestellt. Als Basis dienen dabei die in Kapitel 3 vorgestellten Technologien und das in Kapitel 4 vorgestellte Chemie-Modell der *NAOMI*-Bibliothek.

5.1 Motivation und Ziele

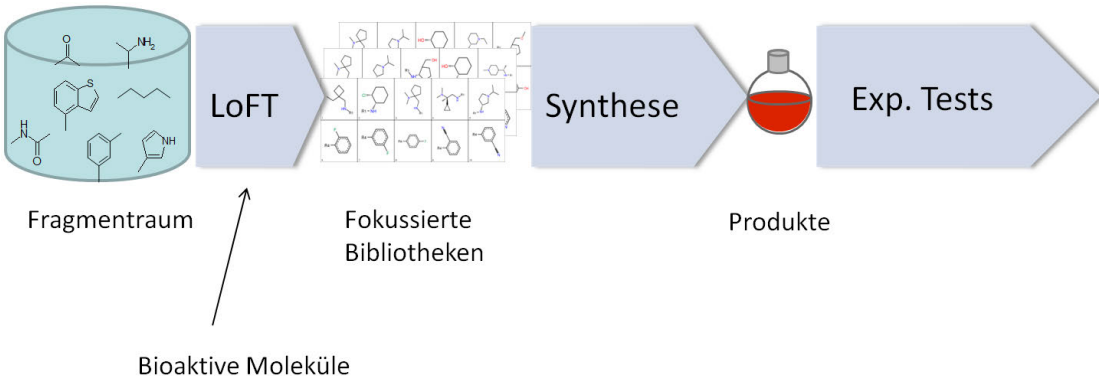


Abbildung 5.1: Grundidee von *LoFT* - Aus einem Fragmentraum, der kombinatorische Bibliotheken beinhaltet, werden fokussierte Bibliotheken generiert. Insbesondere sollen die resultierenden Produkte ähnlich zu bekannten biologisch aktiven Substanzen sein. Anschließend werden die Produkte synthetisiert und experimentell getestet.

Die kombinatorische Chemie führt durch systematische Molekülvariationen zu einer

Vielzahl analoger Verbindungen [16]. Aus Material- oder Kostengründen ist es erforderlich, eine Auswahl der Synthesebausteine anhand vordefinierter Kriterien zu treffen. Die Schwierigkeit liegt folglich in der Aufgabe, bei R Substitutionsstellen eines Grundgerüstes eine von $\prod_{i=1}^{R} \binom{n_i}{k_i}$ möglichen Teilbibliotheken auszuwählen, so dass die betrachteten Kriterien für alle Produkte bestmöglich erfüllt sind. Die Kriterien richten sich nach den Anforderungen des Projektes und sind oftmals gegenläufig. Sind für das Zielprotein bereits aktive Substanzen bekannt, können diese als Vorlage dienen, um ähnliche Moleküle zu generieren. Des Weiteren können Inaktivität, Nebenwirkungen oder patentrechtliche Probleme bestimmter Substanzen den Wunsch hervorrufen, die Optimierung in eine bestimmte Richtung zu lenken. Die Produkte sollen dann möglichst unähnlich zu diesen Molekülen sein. Dabei ist es von Vorteil, wenn das Ähnlichkeitsmaß den Grundgerüstwechsel ermöglicht. Die Auswahl chemisch möglichst unterschiedlicher Reagenzien erlaubt es zudem, einen größeren chemischen Unterraum abzudecken. Dennoch sind Ähnlichkeit und Diversität keine ausreichenden Designkriterien. Vielmehr sollten die physikochemischen Eigenschaften der entstehenden Substanzen einem bestimmten Profil entsprechen, um die Wahrscheinlichkeit zu erhöhen, dass sich diese als Leitstruktur oder Wirkstoff eignen (siehe Kapitel 2.3.1).

Das nachfolgend im Detail vorgestellte Verfahren zum Entwurf fokussierter Bibliotheken wurde entwickelt, um all diese Designziele zu vereinigen. Insbesondere die Ähnlichkeit zu bekannten aktiven Molekülen steht bei LoFT (*Library Optimizer using Feature Trees*) im Vordergrund. Die vorgeschlagenen Bibliotheken werden anschließend synthetisiert und experimentell getestet, beziehungsweise dienen zumindest als Ideengenerator für die Synthese (Abbildung 5.1).

5.2 Arbeitsablauf

Zunächst wird ein Fragmentraum (siehe Kapitel 3.1) eingelesen. Er beinhaltet die Syntheseregeln sowie eine Menge von Reaktanten. Optional können Anfragemoleküle und benutzerdefinierte Eigenschaften eingelesen werden. Zudem können die Reagenzien gefiltert oder in Cluster eingeteilt werden. Mittels Ähnlichkeit, Diversität und additiven physikochemischen Eigenschaften wird eine Zielfunktion für die multikriterielle Optimierung definiert. Anschließend wird die Optimierung gestartet, um fokussierte Bibliotheken zu

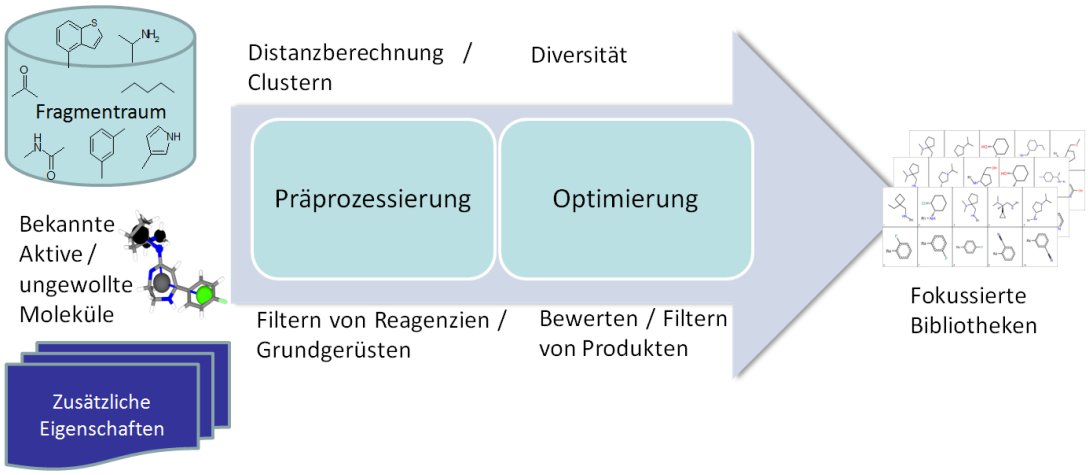


Abbildung 5.2: Arbeitsablauf von *LoFT* - Nach Einlesen eines Fragmentraums und optionaler Anfragemoleküle folgen Präprozessierungsschritte wie das Filtern oder das Clustern der Reagenzien. Anschließend erfolgt die Optimierung, um eine fokussierte Bibliothek zu generieren.

generieren (Abbildung 5.2). Hierfür bietet *LoFT* mehrere stochastische Optimierungsverfahren. Die Auswertung und Weitergabe der Resultate wird mittels verschiedener Ausgabeformate erleichtert. Die Ein- und Ausgabeformate von *LoFT* werden in Anhang B.1 beziehungsweise B.2 näher beschrieben.

5.3 Neuartigkeit

Um die fokussierte Bibliothek in ihrer Gesamtheit zu optimieren, müssen theoretisch alle entstehenden Produkte enumeriert und individuell beurteilt werden. Dafür wird jedes Produkt aus den Fragmenten zusammengebaut und die jeweiligen Deskriptoren werden generiert und bewertet. Das Besondere an der hier vorgestellten Methode ist jedoch, dass die Produkteigenschaften bewertet werden können, obwohl die Berechnung auf Ebene der Reaktanten erfolgt. Durch die Verwendung additiver Molekül-Eigenschaften in Verbindung mit der FTree-Technologie für den Ähnlichkeitsvergleich (Kapitel 3.2 und 5.12), ist eine explizite Generierung der Produkte für die Bewertung nicht notwendig. Stattdessen werden die Eigenschaften der Reaktanten aufaddiert. Die Berechnung der Ähnlichkeit zum Anfragemolekül erfolgt effizient durch Zuordnung der jeweiligen Teilstrukturen in einem baumbasierten Ansatz.

5.4 Anwendungsgebiete

Das Hauptanwendungsgebiet von *LoFT* ist das multikriterielle Design fokussierter Bibliotheken mit besonderem Fokus auf die Ähnlichkeit zu bekannten aktiven Molekülen.

Neben der Auswahl der Reagenzien zur Fokussierung einer Bibliothek kann *LoFT* und insbesondere sein Clustering-Modul (siehe Kapitel 5.13) genutzt werden, um diverse Bibliotheken für das generelle Screening zu generieren.

Des Weiteren kann *LoFT* eingesetzt werden, um die besten Produkte eines Grundgerüstes in Bezug auf die Bewertungsfunktion zu finden (Cherry-Picking). Sollen nur einzelne Moleküle synthetisiert werden, ist das Cherry-Picking ein geeignetes Verfahren zur Auswahl. Eine weitere mögliche Anwendung ist, zunächst vielversprechende Grundgerüste für eine weitere Optimierung zu identifizieren. Im Gegensatz zu FTrees-FS, welches zwar die effiziente Suche nach den ähnlichsten Produkten unter Berücksichtigung von Diversitätskriterien bietet, ist *LoFT* in der Lage, ähnliche Produkte auszuwählen, deren Eigenschaften zusätzlich vorgegebenen Profilen entsprechen.

5.5 Optimierungsverfahren

Für die Optimierung kombinatorischer Bibliotheken werden aufgrund der Größe des Suchraumes Metaheuristiken angewandt (vergleiche auch Kapitel 2.2.1). Die verwendeten Verfahren gehören zu den naturanalogen Verbesserungsverfahren. Sie werden in vielen Gebieten eingesetzt, um kombinatorische Optimierungsprobleme, wie beispielsweise das Problem des Handlungsreisenden, zu lösen.

Beginnend mit einer Anfangsbibliothek wird in jedem Schritt zufällig ein Reagenz ausgetauscht. Daraufhin wird die veränderte Bibliothek nach vorher festgelegten Kriterien bewertet und entweder angenommen oder abgelehnt (siehe Abbildung 5.3). Das Ziel der Optimierung ist dabei, die Güte der fokussierten Bibliothek bezüglich der jeweiligen Zielfunktion zu maximieren.

- Der *Bergsteigeralgorithmus* (*Hill Climbing*) akzeptiert ausschließlich bessere Lösungen. Da der Algorithmus lokale Maxima nicht überwinden kann, ist es notwendig, dass er mehrfach mit unterschiedlichen Startpunkten ausgeführt wird. Er kann verwendet werden, um das Resultat eines vorherigen Optimierungslaufes weiter zu verbessern.

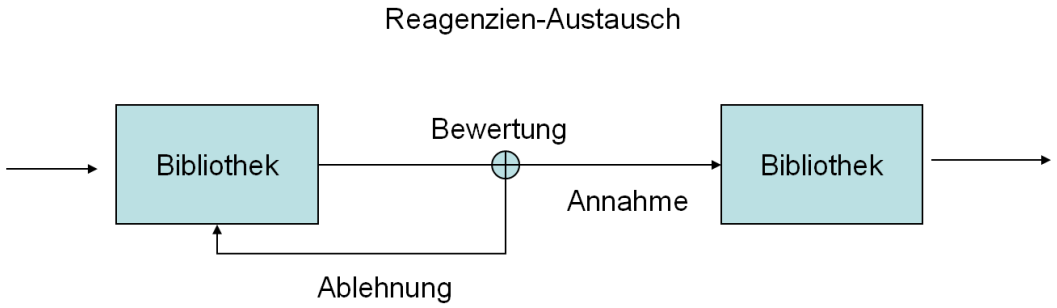


Abbildung 5.3: Optimierungsschritt - Ein Reagenz wird zufällig ausgetauscht. Die neue Bibliothek wird bewertet und entweder angenommen oder abgelehnt.

- *Simulierte Abkühlung* (*Simulated Annealing*) [60] gehört zu den am häufigsten verwendeten Optimierungsverfahren. Dabei wird in Anlehnung an den Abkühlungsprozess von Metallen, ein hoher Temperaturwert kontinuierlich verringert. Ist die Temperatur hoch, werden Lösungen mit schlechterer Bewertung mit höherer Wahrscheinlichkeit erlaubt. Mit Verringerung der Temperatur sinkt die Chance, dass eine schlechtere Lösung akzeptiert wird. Die Berechnung dieser Wahrscheinlichkeit erfolgt mit der natürlichen Exponentialfunktion, wobei sich der Exponent aus ΔE, der Bewertung der derzeitigen Bibliothek minus der Bewertung der neuen Bibliothek, sowie T_t der Temperatur zum Zeitpunkt t zusammensetzt: $e^{(-\frac{\Delta E}{T_t})}$

- Bei der *Schwellenwertakzeptanz* (*Threshold Accepting*) [184] ist eine Verschlechterung der Lösung erlaubt, sofern ΔE unterhalb eines bestimmten Schwellenwertes liegt.

- Der *Sintflutalgorithmus* (*Great Deluge*) [185] simuliert einen steigenden Wasserspiegel. Eine schlechtere Lösung wird angenommen, wenn deren Bewertung über diesem Schwellenwert liegt. Der Wasserspiegel wird während des Optimierungsprozesses angehoben.

Grundsätzlich unterscheiden sich die Verfahren allein anhand des Kriteriums, unter welchen Umständen ein neuer, schlechterer Zustand angenommen wird. Bessere Bibliotheken werden stets angenommen.

LoFT speichert die n besten fokussierten Bibliotheken, die während der Optimierung betrachtet wurden. Dadurch ist zum Beispiel bei der Simulierten Abkühlung ein

stabilisierender Effekt der Abkühlungsfunktion nicht so bedeutend [186], so dass der Optimierungslauf nicht zwangsläufig im lokalen Maximum enden muss. Zudem merkt sich der Algorithmus die Produktbewertungen der letzten Iteration. Da in jedem Schritt genau ein Reagenz ausgewechselt wird, werden lediglich die betroffenen Produkte neu bewertet. Die Bewertungen der Reagenzien (siehe Kapitel 5.6) können anschließend erneuert werden, indem die Bewertungen der neuen Produkte für diese aufaddiert und die der verworfenen Produkte abgezogen werden. Des Weiteren müssen dadurch bei Ablehnung der neuen Bibliothek die Produkte nicht noch einmal bewertet werden.

Neben den Algorithmen zur Generierung fokussierter Bibliotheken wurde ein Cherry-Picking-Algorithmus implementiert, der die n besten Produkte bezüglich der verwendeten Bewertungsfunktion findet.Dafür werden auf Fragment-Ebene alle $\prod_{i=1}^{R} n_i$ möglichen Produkte betrachtet und bewertet. Dies ist aufwändiger als der FTrees-FS-Ansatz [31]. Bei FTrees-FS werden lediglich die k ähnlichsten Fragmente zu jeder Substruktur der Anfrage betrachtet, um ähnliche Moleküle zu generieren. Im Gegensatz dazu ist es bei *LoFT* möglich, ein physikochemisches Profil für die Moleküle zu definieren. Um den Suchraum einzuschränken, können die Reagenzien vorsortiert werden (siehe Kapitel 5.10), so dass nur die jeweils k besten Reagenzien betrachtet werden. Unter Umständen ist es jedoch schwierig, eine Sortierfunktion zu definieren, da von den Eigenschaften der Reagenzien auf die Eigenschaften der Produkte geschlossen werden muss. Empfehlenswert ist es dennoch, zumindest nach Ähnlichkeit zu sortieren, so dass Reagenzien, die sehr unähnlich zu jedweder Teilstruktur der Anfrage sind, aus dem Suchraum ausgeschlossen werden können. Ebenso kann die Menge der zu betrachtenden Reagenzien durch die Definition eines Filters (Kapitel 5.11) eingeschränkt werden.

5.6 Bewertungsfunktionen

Um die Güte einer fokussierten Bibliothek zu bestimmen, wird eine Bewertungsfunktion benötigt (siehe Kapitel 2.2.4). Für die Beurteilung der Reagenzien werden zunächst die Produkte bewertet, was wiederum die Bewertung der Bibliothek erlaubt.

5.6.1 Bewertung eines Produktes

Da die Produkte der Bibliothek vor allem ähnlich zu bekannten biologisch aktiven Referenzmolekülen sein sollen, eignet sich eine gewichtete Bewertungsfunktion der Form $\sum_i^n \omega_i s_i$, wobei bei n Kriterien ω_i das Gewicht und s_i die Bewertung des Kriteriums i ist (siehe Tabelle 5.1a). Wünschbarkeitsfunktionen (*desirability functions*) [64–66] werden genutzt, um makromolekulare Eigenschaften mit unterschiedlichen Maßeinheiten bewerten zu können. Sie werden beispielsweise auch von Le Bailly de Tilleghem et al. [136] und Brown et al. [187] verwendet.

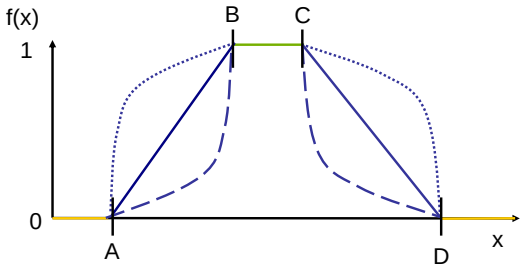


Abbildung 5.4: Bewertung von Eigenschaftswerten - Unter Verwendung von Wünschbarkeitsfunktionen kann ein Eigenschaftswert in das Intervall [0,1] abgebildet werden. Der Wertebereich zwischen den Punkten B und C ist erwünscht und wird mit 1 bewertet. Links von A und rechts von D ist der Wert unerwünscht und wird mit 0 bewertet. Zuletzt muss bestimmt werden, wie die Funktion für die Flanken auszusehen hat.

Bei einer Wünschbarkeitsfunktion wird einem Eigenschaftswert e eine reelle Zahl aus dem Intervall [0,1] zugewiesen, die Wünschbarkeit (siehe Abbildung 5.4). Dadurch kann zwischen erwünschten ($f(e) = 1$), unerwünschten ($f(e) = 0$) und verbesserungsfähigen Werten ($0 < f(e) < 1$) unterschieden werden (Formel 5.1).

In *LoFT* werden die vier Punkte A, B, C und D eines Trapezes festgelegt (siehe Abbildung 5.4). Der Wertebereich zwischen B und C ist erwünscht, links von A und rechts von D sind die Werte unerwünscht (Formel 5.1).

$$f(e) = \begin{cases} 1, & \text{wenn } B \leq e \leq C \\ \frac{(e-A)^n}{(B-A)^n}, & \text{wenn } A < e < B \\ \frac{(D-e)^n}{(D-C)^n}, & \text{wenn } C < e < D \\ 0, & \text{sonst} \end{cases} \tag{5.1}$$

Durch die Skalentransformation ist die resultierende Wünschbarkeit unabhängig von der jeweiligen Maßeinheit. Zudem ist es möglich, für jede Eigenschaft anzugeben, ob

die Funktion langsam ($n = \frac{1}{2}$), schnell ($n = 2$) oder linear ($n = 1$) in Richtung unerwünschter Werte abfallen soll. Für jede Eigenschaft ist die Wünschbarkeitsfunktion separat definierbar.

Die Berechnung der FTree-Ähnlichkeit liefert ebenfalls einen Wert im Intervall [0,1] zurück. Sollen nicht allzu ähnliche Produkte in Bezug auf die Anfrage zu generieren, kann der maximale – und der minimale – Ähnlichkeitswert definiert werden. Ist dieser Schwellenwert über- beziehungsweise unterschritten, ergibt sich ein Ähnlichkeitswert von 0.

5.6.2 Bewertung einer Bibliothek

Für die Bewertung einer Bibliothek werden zunächst die Reagenzien anhand der durchschnittlichen Bewertung der jeweiligen Produkteigenschaften (siehe Tabelle 5.1, Zeile b) beurteilt. Die Bewertung der Bibliothek ist schließlich das arithmetische, geometrische, quadratische Mittel, das Maximum oder Minimum der Reagenzienbewertungen (siehe Tabelle 5.1, Zeile c - g). Der gewählte Modus für die Bewertung der Bibliothek kann die Zusammensetzung der Bibliothek drastisch beeinflussen. Unter Verwendung des Maximums kann die Bibliothek beispielsweise nur einige wenige Reagenzien mit sehr guter Bewertung beinhalten.

5.7 Verwendung mehrerer Anfragemoleküle oder Grundgerüste

Sind zu einem Zielprotein mehrere Inhibitoren bekannt, kann es von Interesse sein, Produkte zu generieren, die eine gewisse Ähnlichkeit zu diesen Inhibitoren aufweisen oder deren physikochemischen Eigenschaften kombinieren. Wird eine Bibliothek auf mehrere Anfragemoleküle fokussiert, ist entweder der maximale Ähnlichkeitswert (siehe Tabelle 5.1, Zeile h) oder der durchschnittliche Ähnlichkeitswert (siehe Tabelle 5.1, Zeile i) zur Bewertung eines Produktes in Bezug auf die Anfragemoleküle verwendbar. Analog gilt dies für Moleküle, zu denen die Produkte unähnlich sein sollen (siehe Tabelle 5.1, Zeile j und k). Zu beachten ist, dass der FTree-Vergleich der laufzeitrelevante Schritt bei der Optimierung ist. So führt die Hinzunahme eines zweiten, strukturell ähnlichen Moleküls etwa zu einer Verdopplung der Laufzeit.

Formel	Beschreibung
a) $pscore(p) = \sum_{i=1}^{\lvert d\rvert} \omega_i s_i$	Zur Bewertung eines Produktes p wird die gewichtete Summe über eine Deskriptormenge d berechnet, wobei ω_i die Gewichtung und s_i die Bewertung des i-ten Deskriptors ist. Dabei gilt: $\sum_{i=1}^{\lvert d\rvert} \omega_i = 1$
b) $rscore(r) = \frac{1}{\lvert P(r)\rvert} \sum_{p \in P(r)} pscore(p)$	Die Bewertung eines Reagenzes r ergibt sich aus der durchschnittlichen Produktbewertung, dabei gilt $P(r) = \{Produkte\ p \mid p\ beinhaltet\ r\}$.
c) $lscore_{arith}(b) = \frac{1}{\lvert R(b)\rvert} \sum_{r \in R(b)} rscore(r)$	Die Bewertung einer Bibliothek b ergibt sich aus dem arithmetischen Mittel über die Reagenzienbewertungen, dabei gilt $R(b) = \{Reagenzien\ r \mid b\ beinhaltet\ r\}$.
d) $lscore_{geo}(b) = \left(\prod_{r \in R(b)} rscore(r)\right)^{1/\lvert R\rvert}$	Die Bewertung einer Bibliothek b ergibt sich aus dem geometrischen Mittel über die Reagenzienbewertungen, dabei gilt $R(b) = \{Reagenzien\ r \mid b\ beinhaltet\ r\}$.
e) $lscore_{rms}(b) = \sqrt{\frac{1}{\lvert R(b)\rvert} \sum_{r \in R(b)} rscore(r)^2}$	Die Bewertung einer Bibliothek b ergibt sich aus dem quadratischen Mittel über die Reagenzienbewertungen, dabei gilt $R(b) = \{Reagenzien\ r \mid b\ beinhaltet\ r\}$.
f) $lscore_{min}(b) = min(rscore(r)) \mid r \in R(b))$	Die Bewertung einer Bibliothek b ergibt sich aus der minimalen Reagenzienbewertung, dabei gilt $R(b) = \{Reagenzien\ r \mid b\ beinhaltet\ r\}$.
g) $lscore_{max}(b) = max(rscore(r)) \mid r \in R(b))$	Die Bewertung einer Bibliothek b ergibt sich aus der maximalen Reagenzienbewertung, dabei gilt $R(b) = \{Reagenzien\ r \mid b\ beinhaltet\ r\}$.
h) $qsim_{max}(p) = max(sim(i,p) \mid i \in q)$	Maximale Ähnlichkeit eines Produktes p zu jedem Anfragemolekül i aus einer Menge von Anfragemolekülen q.
i) $qsim_{avg}(p) = \frac{1}{\lvert q\rvert} \sum_{i=1}^{\lvert q\rvert} sim(i,p)$	Durchschnittliche Ähnlichkeit eines Produktes p zu einer Menge von Anfragemolekülen q.
j) $asim_{max}(p) = 1 - max(sim(i,p) \mid i \in a)$	Maximale Unähnlichkeit eines Produktes p zu jedem unerwünschten Molekül i aus der Menge a.
k) $asim_{avg}(p) = \frac{1}{\lvert a\rvert} \sum_{i=1}^{\lvert a\rvert}(1 - sim(i,p))$	Durchschnittliche Unähnlichkeit eines Produktes p zu den unerwünschten Molekülen der Menge a.

Tabelle 5.1: Die Bewertungsfunktionen von *LoFT* und ihre Beschreibung.

Des Weiteren kann eine Bibliothek mit mehreren Grundgerüsten optimiert werden. Hierbei gilt die Einschränkung, dass die Grundgerüste die gleichen Linktypen besitzen müssen. Genauso wie bei der Bewertung der Anfragemoleküle können die Grundgerüste individuell gewichtet werden, um den Einfluss auf die Bibliothek zu kontrollieren. Ebenso kann der maximale Ähnlichkeitswert verwendet werden. Da mehr Ähnlichkeitsvergleiche berechnet werden müssen, ist genauso wie bei der Verwendung von mehreren Anfragemolekülen eine höhere Laufzeit zu erwarten. Neben den zusätzlichen Ähnlichkeitsvergleichen müssen gleichermaßen alle weiteren Deskriptoren unter Berücksichtigung des zusätzlichen Grundgerüstes evaluiert und bewertet werden.

5.8 Bibliotheken ohne Grundgerüst

Neben den kombinatorischen Bibliotheken mit Grundgerüst sind insbesondere zwei weitere Bibliothekstypen von Interesse: R_1-R_2-Bibliotheken und R_1-R_2-R_3-Bibliotheken. Für erstere wurde ein Pseudo-Grundgerüst eingeführt, also ein Fragment, welches aus zwei kompatiblen Linkatomen besteht. Nach der Definition der beiden Linktypen durch den Nutzer wird automatisch der Reagenzien-Filter (siehe Kapitel 5.11) so erweitert, dass ausschließlich Reagenzien mit diesen Linktypen erlaubt sind. R_1-R_2-R_3-Bibliotheken können bisher nicht optimiert werden, da hierfür der FTree-Vergleich (siehe Kapitel 5.12) umgestellt werden müsste. Behelfsmäßig kann eine Teilenumeration erfolgen.

5.9 Deskriptoren

Für die Bewertung der Reaktanten und Produkte werden unterschiedliche Deskriptoren verwendet. Dabei muss unterschieden werden zwischen Deskriptoren, die einzig auf der Struktur eines Moleküls beruhen, und solchen, die von dem Verhältnis zu anderen Molekülen, Proteinen oder der Umgebung abhängig sind. Für die Bewertung dieser Art von Deskriptoren sind komplexere Berechnungen notwendig [23]. Bei der Verwendung von Fragmenträumen ist zusätzlich eine weitere Unterteilung erforderlich. Hier ist von Interesse, welche Deskriptoren die Berechnung der Produkteigenschaften durch die Berechnung mittels der Fragmente zumindest approximieren. So kann der Vergleich zweier Moleküle durch die FTree-Technologie auf Fragmentebene approximiert werden (siehe Kapitel 3.2.4 und 5.12).

Die Deskriptoren (Tabelle 5.2) werden nachfolgend in vier Kategorien eingeteilt. Die Einteilung erfolgt im Hinblick auf kombinatorische Bibliotheken und nicht für den allgemeinen Fall von Fragmenträumen. Eine zusätzliche Voraussetzung ist, dass durch die Verbindungsregeln des Fragmentraumes keine Ringschlüsse modelliert werden.

- *Additive Deskriptoren.* Die Deskriptoren können über die Fragmente aufsummiert werden. Beispielhaft sind die Anzahl der Schweratome oder Ringsysteme zu erwähnen.

- *Korrigierbare Deskriptoren.* Prinzipiell sind diese Deskriptoren additiv. Durch bestimmte Verknüpfungsregeln kann sich die Bindungsordnung und dadurch der Protonierungszustand und Atomtyp der neu verbundenen Atome ändern. Ein Beispiel ist die Anzahl der Wasserstoffbrücken-Akzeptoren. Solange kein Pseudo-Grundgerüst (siehe Kapitel 5.8) genutzt wird, ist es möglich, einen Korrekturmechanismus anzuwenden (siehe Kapitel 5.9.1). Ein Sonderfall ist der längste Pfad rotierbarer Bindungen (*contiguous rotatable bonds*, CRTB). Er benötigt zur korrekten Berechnung die Speicherung zusätzlicher Daten. Neben dem längsten Pfad im Fragment wird für jedes Linkatom der längsten Pfad ins Fragment sowie zu den anderen Linkatomen abgespeichert. Dadurch können bei der Verknüpfung zweier Fragmente die korrekten Werte für das neue Fragment berechnet werden.

- *Approximative Deskriptoren.* Zu dieser Kategorie gehören Deskriptoren, die von Substrukturen abhängig sind, welche sich über mehrere Fragmente erstrecken können. Wie gut sich die Deskriptoren approximieren lassen, hängt vom molekularen Graphen der verwendeten Grundgerüste ab. Das gilt zum Beispiel für SMARTS, Stereodeskriptoren und die Anzahl der Vorkommnisse einer bestimmten funktionellen Gruppe. Des Weiteren ist der Feature-Tree-Deskriptor approximativ, da die initialen Schnitte lediglich im Feature-Tree des Grundgerüstes erfolgen.

- *Nicht-approximative Deskriptoren.* Für diese Deskriptoren existieren im Kontext von LoFT keine geeigneten Verfahren auf Fragment-Ebene. Ein Beispiel ist die 3D-Überlagerung von Produkt und Anfragemolekül, die zur Auswahl der Reagenzien und zum Filtern der Produkte verwendet werden kann (Kapitel 5.14).

Im Folgenden soll auf den Korrekturmechanismus (Kapitel 5.9.1) eingegangen werden.

Bezeichner	Typ	Wertebereich	Beschreibung
MW	korrigierbar	$[0, \infty]$	Die Summe der Atommassen aller Atome (Molekulargewicht)
Volume	korrigierbar	$[0, \infty]$	Die Summe der Atomvolumina
TPSA	korrigierbar	$[0, \infty]$	Methode zur Berechnung der polaren Oberfläche anhand von Fragmenten [114]
PLogP	korrigierbar	$[-\infty, \infty]$	Der Logarithmus des Verteilungskoeffizienten eines Stoffes in zwei nicht mischbaren Phasen (Oktanol und Wasser) im Gleichgewicht. Die Summe der partiellen Beiträge der einzelnen Atomtypen ist ein Maß für die Hydrophobizität.
Atoms	additiv	$[0, \infty]$	Die Anzahl der Atome, die weder Wasserstoff noch Linkatom sind (Schweratome)
Hetero	additiv	$[0, \infty]$	Die Anzahl der Heteroatome, die weder Wasserstoff, Kohlenstoff noch Linkatom sind
NO	additiv	$[0, \infty]$	Die Anzahl der Sauerstoffe und Stickstoffe
Linkers	korrigierbar	$[0, \infty]$	Die Anzahl der Linkatome
Acceptors	korrigierbar	$[0, \infty]$	Die Anzahl der Wasserstoffbrückenakzeptoren
Donors	korrigierbar	$[0, \infty]$	Die Anzahl der Wasserstoffbrückendonoren
AromAtoms	korrigierbar	$[0, \infty]$	Die Anzahl der aromatischen Atome
Halogens	additiv	$[0, \infty]$	Die Anzahl der Halogenatome (F, Cl, Br, I, At)
Inorganic	additiv	$[0, \infty]$	Die Anzahl der Atome, die nicht organisch sind
Charge	korrigierbar	$[-\infty, \infty]$	Die Formalladung des Moleküls
Bonds	additiv	$[0, \infty]$	Die Anzahl der kovalenten Bindungen zwischen Schweratomen
Rotb	korrigierbar	$[0, \infty]$	Die Anzahl der rotierbaren Bindungen
CRTB	korrigierbar	$[0, \infty]$	Der längste, kontinuierliche Pfad rotierbarer Bindungen ist ein Maß für die Flexibilität des Moleküls.
Ringsystems	additiv	$[0, \infty]$	Anzahl der Ringsysteme
AroRingsystems	additiv	$[0, \infty]$	Anzahl der vollständig aromatischen Ringsysteme

Tabelle 5.2: Verwendete Moleküleigenschaften. Mittels des Bezeichners lässt sich die Eigenschaft in *LoFT* ansteuern.

Bezeichner	Typ	Wertebereich	Beschreibung
MaxRing	additiv	$[0, \infty]$	Größter Ring in Bezug auf die Anzahl der Schweratome
MaxRSsize	additiv	$[0, \infty]$	Größtes Ringsystem in Bezug auf die Anzahl an Schweratomen
Rings	additiv	$[0, \infty]$	Anzahl der relevanten Ringe [188]
AroRings	additiv	$[0, \infty]$	Anzahl der aromatischen Ringe
RS	approximativ	$[0, \infty]$	Anzahl der R/S Stereozentren
EZ	approximativ	$[0, \infty]$	Anzahl der E/Z Stereobindungen
SMARTS	nicht-approximativ	$[0, \infty]$	Anzahl der Vorkommnisse eines bestimmten Subgraphen
Func	nicht-approximativ	$[0, \infty]$	Anzahl der Vorkommnisse einer funktionellen Gruppe: Der Name der Gruppe erlaubt den Zugriff auf die interne Molekülannotation.
LinkName	nicht-approximativ	$[0, \infty]$	Anzahl der Vorkommnisse eines Linknamens zum Filtern (siehe Kapitel 5.11)
Frei wählbar	additiv	$[-\infty, \infty]$	Vom Nutzer hinzugefügte Eigenschaften, die strikt additiv sein sollen.

Tabelle 5.2: - weitergeführt - Die verwendeten Moleküleigenschaften

5.9.1 Deskriptor-Korrektur

Soll von additiven Fragmenteigenschaften auf die Eigenschaften der Produkte geschlossen werden, ist zu beachten, dass sich durch die Anwendung der Verknüpfungsregeln die Bindungsordnung der zu den Linkatomen adjazenten Atomen ändern kann. Beispielsweise ist dies der Fall, wenn die Linkatome mit einer Einfachbindung an den Bindungspartner gebunden sind, die Verknüpfungsregel aber zu einer Doppelbindung führt. In diesem Fall werden Wasserstoffe an den zu verbindenden Atomen entfernt, wodurch sich die Eigenschaftswerte ändern.

Werden die Produkte anhand der additiven Fragmenteigenschaften gefiltert oder bewertet, wie beispielsweise beim Fragmentraum-Enumerator [68], kann dies zur Akzeptanz oder Ablehnung von Produkten mit den falschen Eigenschaftswerten führen. Für kombinatorische Bibliotheken lässt sich dieses Problem hingegen bei den als korrigierbar bezeichneten Deskriptoren lösen. Hierfür werden Grundgerüst und Reagenz verbunden und die Deskriptoren berechnet. Anschließend werden die Eigenschaftswerte des Grundgerüstes abgezogen. Die Eigenschaftswerte werden für das jeweilige Reagenz

gespeichert.

Während der Optimierung werden die Werte verwendet, so dass sich durch Aufsummierung die korrekten Produktwerte ergeben (siehe Abbildung 5.5). Lediglich bei der Verwendung eines Pseudo-Grundgerüstes können die Werte noch abweichen. Die Vorhersage der approximativen Eigenschaften ist nicht in jedem Fall korrekt. Die Stereozentren lassen sich beispielsweise nur bestimmen, wenn der vollständige Molekülgraph bekannt ist, da unter Umständen die Atomprioritäten nicht aufgelöst werden können. Dennoch führt die Korrektur zu einer besseren Näherung.

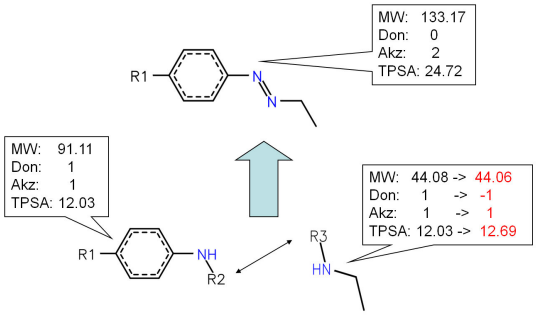


Abbildung 5.5: Korrektur der Deskriptoren - Grundgerüst (links) und Reagenz (rechts) werden verbunden und die Deskriptoren berechnet. Dann werden die Werte des Grundgerüstes abgezogen und für den Reagenz gespeichert.

5.10 Vorauswahl und Sortierung der Reagenzien

Die Leitstrukturoptimierung ist ein zyklischer Vorgang [189]. Oftmals ist es deshalb wünschenswert, dass bestimmte Reagenzien in einer weiteren Optimierung mit anderen oder zusätzlichen Kriterien erhalten bleiben. Ebenso kann die Vergrößerung der bestehende Bibliothek eine Anforderung sein. Deshalb ist es vor der Optimierung möglich, Reagenzien zu selektieren, die in die Startbibliothek gewählt werden und während der Optimierung nicht ausgetauscht werden dürfen. Für jedes Linkatom des Grundgerüstes können kompatible Reagenzien vorselektiert werden.

Zudem ist es möglich, die Reagenzien mit Hilfe der in Kapitel 5.6 beschriebenen Bewertungsfunktion zu sortieren. Die Sortierung ist unter anderem in zwei Szenarien hilfreich. Zum einen bilden die Reagenzien mit der besten Bewertung einen guten Ausgangspunkt für die stochastische Optimierung, insbesondere bei der Optimierung in Bezug auf die Ähnlichkeit zu einer Anfrage. Gesetzt den Fall, es soll nur eine einzelne Eigenschaft maximiert beziehungsweise minimiert werden, ist es möglich durch die Sortierung der Reagenzien bereits eine optimale Lösung zu erreichen. Zum anderen kann durch die Vorsortierung der Suchraum eingeschränkt werden, indem nur die

besten n Reagenzien für jedes Linkatom des Grundgerüstes betrachtet werden. Dies ist insbesondere beim Cherry-Picking von Vorteil. Da die Reagenzien ein Fragment der Produkte sind, ist es erforderlich, die erlaubten Eigenschaftsbereiche anzupassen. Werden die Reagenzien anhand ihrer Ähnlichkeit zu einem Anfrage-Molekül sortiert, ist es unter Umständen hilfreich, das Grundgerüst mitzubetrachten. Für die Ähnlichkeit zur Anfrage gilt außerdem, dass bei der Sortierung das Reagenz auf sämtliche Teilbäume des Anfrage-FTrees gelegt und die Ähnlichkeit zu diesen berechnet wird. Anschließend geht der maximale Ähnlichkeitswert in die Bewertung des Reagenzes ein. Dabei werden die in Kapitel 5.12.2 beschriebenen Restriktionen beachtet.

5.11 Filterung und statistische Analyse von Molekülmengen anhand ihrer Eigenschaften

Sowohl Reagenzien als auch Grundgerüste können mit Filtern selektiert werden. Daraus ergibt sich die Möglichkeit, den Eingabe-Suchraum für die Optimierung einzuschränken. Dies ist insbesondere von Vorteil, wenn die Reagenzien zum Beispiel einen Wasserstoffbrückendonor beinhalten müssen, damit eine bestimmte Wechselwirkung überhaupt ausgebildet werden kann. Ein Filter kann zusätzlich zur Bewertungsfunktion gewählt werden. Entspricht ein Produkt nicht den Anforderungen, ist es nicht mehr erforderlich, die Bewertungsfunktion anzuwenden. Dies heißt im Speziellen, dass keine laufzeitrelevanten Ähnlichkeitsvergleiche ausgeführt werden müssen. Das Produkt wird stattdessen mit 0 bewertet.

Ein Filter wird definiert, indem für eine Menge von Eigenschaften jeweils ein Minimal- und ein Maximalwert angegeben wird. Diese Operanden werden mit logischen Operatoren verknüpft. Dabei kann die Auswertungsreihenfolge der Operatoren durch Klammerung der Teilausdrücke verändert werden. Zudem wurde ein weiteres Sprachkonstrukt eingeführt, um komplexere Filterausdrücke auf einfache Art zu definieren.

Der Term "$TOLERATE[x]\{exp_1, .., exp_n\}$" erlaubt es, dass x von n Ausdrücken (wobei $x < n$ gilt) als *falsch* ausgewertet werden. Dadurch kann zum Beispiel die *Rule-of-Five* [106] zum Abschätzen der oralen Bioverfügbarkeit folgendermaßen als Filter angewendet werden:

$$TOLERATE[1]\{Donors[0,5],\ Acceptors[0,10],\ MW[0,500],\ pLogP[-100,5]\}$$

Ein weiteres Beispiel ist der folgende Ausdruck, der ausschließlich Moleküle mit einem Molekulargewicht im Bereich von 200 bis 400 Dalton oder mit positiver Formalladung erlaubt:

$$MW[200, 400] \; or \; not \; Charge[-100, 0]$$

Der nachstehende Ausdruck erlaubt Moleküle, die exakt eine Hydroxyl-Gruppe als Substruktur (SMARTS [72]) oder ein Ringsystem und 5 bis 10 Heteroatome enthalten:

$$SMARTS['[OX2H]'][1, 1] \; or \; (Ringsystems[1, 1] \; and \; Hetero[5, 10])$$

Durch die Bereitstellung der logischen Verknüpfungen können in einem Filterausdruck unter anderem verschiedene Filterkriterien für die Reagenzien mit unterschiedlichen Linktypen verwendet werden. So muss für die jeweilige Aufgabe nur ein einziger Filter definiert werden. Ein ähnliches Konzept verwendet auch Colibri [190].

Des Weiteren ist es möglich, das Eigenschaftsprofil einer Molekülmenge zu generieren, um sie statistisch auszuwerten. Ein solches Profil enthält für alle Eigenschaften den Minimal-, Maximal- und Mittelwert, die Standardabweichung, sowie den Median, das erste und das dritte Quartil. Dies ist hilfreich, um Filter für einen Fragmentraum zu definieren und um fokussierte Bibliotheken oder Produkte auswerten und vergleichen zu können. Anhang B.2 zeigt exemplarisch eine Ausgabedatei, die solche Profile für eine generierte Bibliothek enthält. Für detaillierte Auswertungen und Visualisierungen mit externen Programmen, besteht zudem die Möglichkeit, die einzelnen Eigenschaftswerte herauszuschreiben (siehe Anhang B.2).

Die derzeit verwendbaren Eigenschaften und Deskriptoren sind in Tabelle 5.2 aufgeführt. Ebenso können benutzerdefinierte Eigenschaften aus externen Dateien eingelesen werden (siehe Anhang B.1). Der *NAOMI*-Konverter [35] und QSearch [158] verwenden ebenfalls dieses Modul.

5.12 FTree-Ähnlichkeitsvergleich

Analog zu der Idee von FTrees-FS [31] (Kapitel 3.2.4), kann die FTree-Ähnlichkeit zwischen Anfrage und Produkt auf Basis der Fragmente berechnet werden. Bei kombinatorischen Bibliotheken vereinfacht sich die Problemstellung, da durch das Grundgerüst das jeweilige Basisfragment vorgegeben ist. Zunächst wird jedoch die Ähnlichkeit zwischen Anfragesubstruktur und den betreffenden Reagenzien vorberechnet. Damit auf

diese Vergleichswerte während der Berechnung der Ähnlichkeit von Anfrage und Produkt zugegriffen werden kann, werden sie in die Matrix von Anfrage und Grundgerüst eingetragen (siehe Abbildung 5.6). Anschließend wird der Vergleich von Anfrage und Grundgerüst gestartet. Die Ähnlichkeit von Anfrage und Produkt ergibt sich aus der maximalen Ähnlichkeit aller Teilbaumkombination von Anfrage und Grundgerüst.

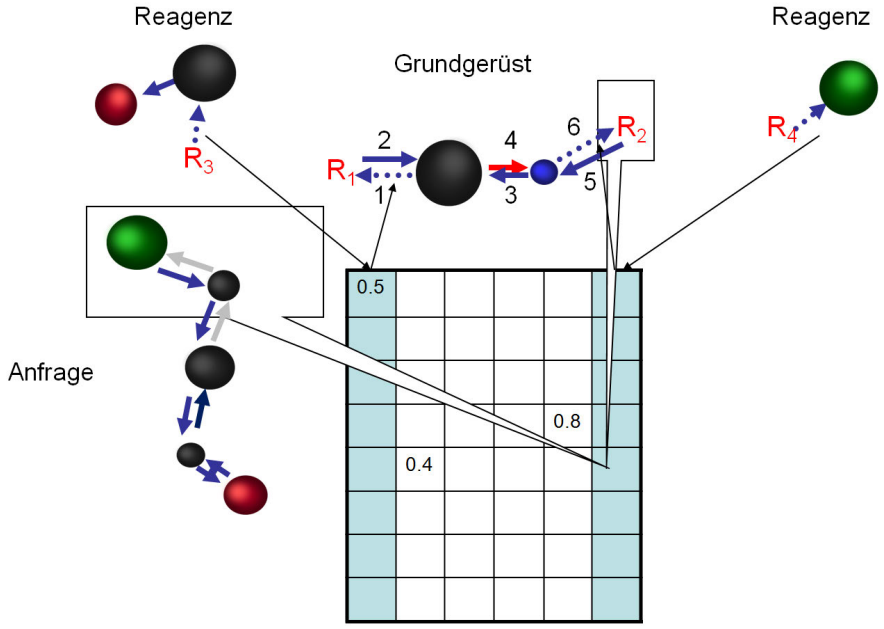


Abbildung 5.6: FTree-Vergleich für kombinatorische Bibliotheken - Die Abbildung zeigt die Matrix für den Vergleich von Anfrage und Grundgerüst. Jede ungerichtete Kante wird durch zwei gerichtete antiparallele Kanten repräsentiert. Durch den Schnitt einer ungerichteten Kante entstehen somit zwei disjunkte, gewurzelte Teilbäume. Die Kanten sind zufällig nummeriert und die Linktypen R1 und R3, beziehungsweise R2 und R4 sind kompatibel. In dem gezeigten Beispiel wurden die ungerichteten Kanten geschnitten, die zu den Teilbäumen führen, deren Wurzelkante rot markiert ist. Der korrespondierende Ähnlichkeitswert wird in die markierte Zelle geschrieben. Um diesen Wert zu berechnen, sind die blauen Zellen, die zu den Linkkanten des Grundgerüst-FTrees gehören (gepunktete Linie), mit den korrespondierenden Werten von Anfrageteilbaum und Reagenz gefüllt worden.

Angewandt auf kombinatorische Bibliotheken kann die Berechnung weiter optimiert werden. Dafür werden die Vergleichswerte von Anfrage und Reagenz in einer globalen Matrix zwischengespeichert, so dass sie nur einmal berechnet werden müssen. Dies ist in Abbildung 5.7 illustriert. Ein solcher zusätzlicher Zwischenspeicher wurde bereits zur

Beschleunigung des Feature-Tree-Vergleiches mittels Indexierung eingesetzt [166].

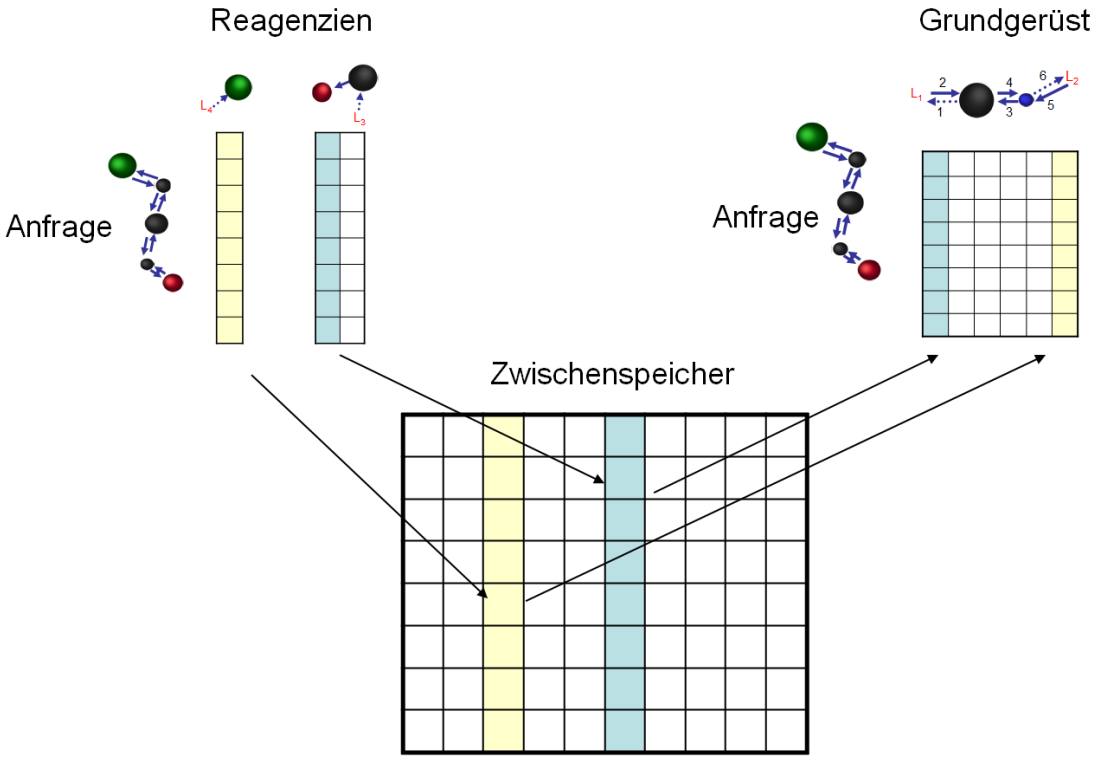


Abbildung 5.7: Wiederverwendung von bereits berechneten Vergleichswerten - Die Ähnlichkeitsvergleiche der FTree-Teilbäume von Anfrage und Reagenz werden verwendet, um die Matrix des Vergleiches von Anfrage und Grundgerüst vorzufüllen. Diese Werte können in einer globalen Matrix zwischengespeichert werden. Die Berechnung muss dadurch nicht wiederholt werden.

Der Zugriff auf den Zwischenspeicher erfolgt über die ID des Anfragemoleküles und eine eindeutige ID, die dem Reagenz zugewiesen wurde. Werden sehr große Fragmenträume verwendet, kann der Zwischenspeicher ausgeschaltet werden. In diesem Fall müssen die Berechnungen stets aufs Neue erfolgen.

Eine weitere Laufzeitverbesserung lässt sich erzielen, indem lediglich ein Teil der Vergleiche neu ausgeführt wird. Da während der Bibliotheksoptimierung in jedem Schritt genau ein Reagenz ausgetauscht wird, müssen bei der Bewertung der fokussierten Bibliothek nur die Produkte neu bewertet werden, die das entsprechende Reagenz beinhalten (siehe Kapitel 5.5). Dabei müssen nicht alle Zellen der Vergleichsmatrix neu berechnet werden. Die Idee ist die folgende: Unterscheiden sich zwei Produkte nur anhand eines

Reagenzes, werden nur die Zellen der Matrix neu berechnet, die auf der Berechnung der zu diesem Reagenz gehörenden Zellen basieren (siehe Abbildung 5.8). Dadurch kann im Falle von Grundgerüsten mit zwei Linkatomen ($k = 2$) eine 50%-Wiederverwendung der berechneten Zellen erzielt werden.

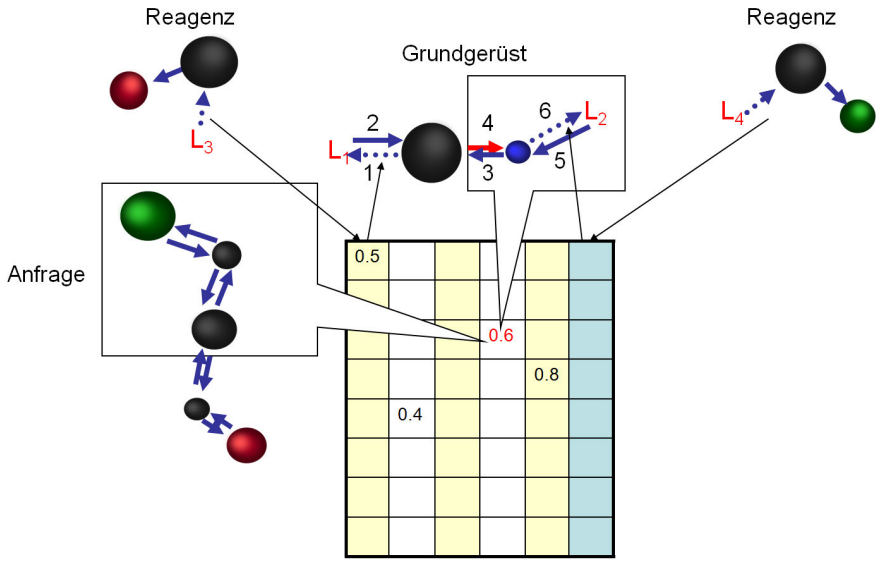


Abbildung 5.8: Wiederverwendung von Zellen - Die Abbildung folgt auf den Vergleich aus Abbildung 5.6 und zeigt den Vergleich für ein weiteres Produkt. Wird während der Enumerierung der Produkte der fokussierten Bibliothek ein Reagenz ausgetauscht, müssen nur die Zellen neu berechnet werden, die auf dem Vergleich dieses Reagenzes beruhen. In diesem Beispiel wurde das rechte Reagenz ausgetauscht. Daraus folgt, dass die gelben Zellen wiederverwendet werden können. Die blauen Zellen müssen dagegen neu gefüllt werden. Anschließend folgt eine Neuberechnung der darauf basierenden (weißen) Zellen.

Des Weiteren wurden Größenfilter eingeführt, die die Optimierung beschleunigen, da bei Ablehnung der Ähnlichkeitsvergleich nicht durchgeführt wird. Die Filter können sowohl auf Reagenzien- als auch auf Produktebene verwendet werden. Dabei wird, wie auch bei FTrees-FS, angegeben, wie groß das Produkt in Bezug auf das Anfragemolekül beziehungsweise das Reagenz in Bezug auf die betrachtete Substruktur sein darf. Die Größe einer Substruktur bestimmt sich über die Anzahl der Schweratome. Für die Produkte gilt:

$$atoms(a) \cdot x_{min} \leq atoms(p) \leq atoms(a) \cdot x_{max} \tag{5.2}$$

In der Formel steht a für das Anfragemolekül und p für das Produkt, sowie x_{min}

und x_{max} für den jeweiligen Größenfaktor. Liegt die Größe des Produks außerhalb des erlaubten Bereiches, wird ein Ähnlichkeitwert von 0 zurückgegeben, wodurch sich der Ähnlichkeitsvergleich erübrigt. Analog gilt die Formel für Reagenzien. Hier wird der Wert von 0 im Zwischenspeicher abgelegt.

5.12.1 Erweiterungen des Vergleichsverfahrens

Während der Evaluation des ersten Prototypen von *LoFT* [32] wurden zwei Nachteile der verwendeten FTree-Deskriptoren offenkundig. Diese haben beim Design fokussierter Bibliotheken einen großen Einfluss:

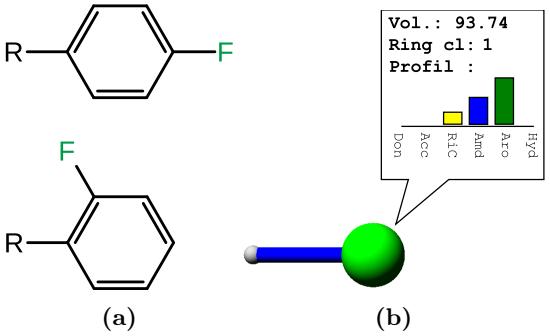


Abbildung 5.9: Identischer Feature-Tree für Regioisomere - Sowohl para-Fluorophenyl (a, oben) als auch ortho-Fluorophenyl (a, unten) ergeben den identischen Feature-Tree (b).[2]Dadurch können diese Regioisomere beim FTree-Vergleich nicht unterschieden werden.

- In der Phase der Leitstrukturidentifizierung ist ein generischer Ansatz zur Bestimmung molekularer Ähnlichkeit beziehungsweise Unähnlichkeit erforderlich. Die automatische Zuordnung der FTree-Knoten erlaubt die Auswahl von Bibliotheken und Substanzen für weitere Tests. Ist dagegen das Ziel, eine biologisch aktive Struktur zu optimieren, erwartet der Nutzer zumeist eine spezifische Zuordnung in Bezug auf die Struktur der Anfrage. Dabei kann die automatische Zuordnung der Knoten zu einer unerwünschten Ausrichtung der Feature-Trees führen, so dass das Wissen des Nutzers über den Bindungsmodus des Anfragemoleküls oder die

[2]Die griechischen Vorsilben *ortho* (1,2), *meta* (1,3) und *para* (1,4) beschreiben bei einem aromatischen Sechsring die Position des zweiten Substituenten in Bezug auf den ersten.

Platzierung des Grundgerüstes nicht beachtet wird. Ebenso kann sich die Platzierung und Orientierung des Grundgerüstes mit der Auswahl der Reagenzien ändern (siehe Abbildung 5.10). Insbesondere ist dies der Fall, wenn mehrere mögliche Zuordnungen existieren und zusätzliche Kriterien erfüllt werden sollen. In diesem Fall möchte man die gefundene Zuordnung fixieren. Genauso kann es sein, dass der Benutzer ein alternatives Matching erreichen möchte. Aus diesem Grund wurde eine Möglichkeit geschaffen, wie der Benutzer das FTree-Matching steuern kann. Im Gegensatz zu reaktantbasierten Ansätzen ist es nicht erforderlich, die genaue Position des Grundgerüst zu fixieren. Stattdessen können Bereiche (FTree-Kanten der Anfrage) angegeben werden, denen bestimmte Reagenzien zugeordnet werden dürfen.

- Eine Unterscheidung von regioisomeren Substrukturen[3] auf Ebene des FTree-Deskriptors war bisher nicht möglich. Dies ist beim Entwurf fokussierter Bibliotheken aus zwei Gründen problematisch. Ist bekannt, dass ein bestimmtes Substitutionsmuster relevant für die Bindung zum Protein ist, sollte sich das in der Auswahl der Reagenzien widerspiegeln. Die Fragmente mit unterschiedlichen Substitutionsmustern erhalten jedoch den gleichen Ähnlichkeitswert. Sie werden deshalb zufällig selektiert. Um dies zu vermeiden, wurde das Vergleichsverfahren so erweitert, dass Substitutionsmuster an aromatischen Ringen unterscheidbar sind (siehe Kapitel 5.12.3).

Die Möglichkeit, den Vergleichsalgorithmus direkt zu beeinflussen, ist einer Prä- oder Postprozessierung der Bibliothek unter Verwendung zusätzlicher Deskriptoren vorzuziehen. Ohne die Information über die gefundene Ähnlichkeit zu verlieren, kann der Nutzer von einem generellen Design auf ein spezifisches übergehen. Im Folgenden sollen die Erweiterungen des Ähnlichkeitsvergleiches beschrieben werden, die zur Lösung dieser Probleme entworfen wurden [33].

Bei der Entwicklung waren mehrere Randbedingungen zu beachten: Die Unterscheidung regioselektiver Substrukturen soll sowohl beim paarweisen Vergleich als auch bei der Fragmentraumsuche möglich sein. Zudem soll der Deskriptor weiterhin unabhängig von der Molekülkonformation bleiben. Das heißt, die Unterscheidung muss auch mit

[3]Regioisomere besitzen zwar die gleiche Summenformel, haben jedoch eine andere Struktur (siehe Abbildung 5.9 für ein Beispiel).

2D-Koordinaten realisierbar sein. Ebenso sollen die bestehenden Vergleichsalgorithmen trotz der Erweiterungen möglichst unverändert bleiben, um Seiteneffekte zu vermeiden.

5.12.2 Restriktives Matching

Für die Vorgabe der möglichen Platzierungen des Grundgerüstes spezifiziert der Nutzer, welche Reagenzien-Linktypen auf welche Kanten des Anfrage-FTrees gelegt werden dürfen. Wie in Kapitel 5.12 beschrieben, werden die Zellen der Vergleichsmatrix gefüllt. Allerdings wird für die Reagenz-Linktyp/Anfrage-FTree Kante Kombinationen, welche nicht erlaubt sind, ein hoher negativer Ähnlichkeitswert gespeichert (siehe Abbildung 5.11).

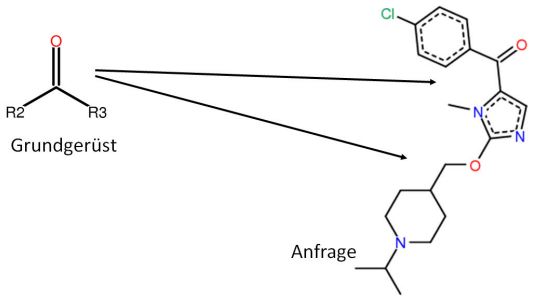


Abbildung 5.10: Alternative Grundgerüstplatzierungen - Je nach Art und Größe des Grundgerüstes ergeben sich unter Umständen mehrere Platzierungsmöglichkeiten.

Anschließend wird die Berechnung des Ähnlichkeitswertes begonnen, wie in Kapitel 5.12 beschrieben. Aufgrund der negativen Werte in den Zellen, wird der Algorithmus dazu forciert, eine bessere Zuordnung zu suchen. Wurde ein Matching gefunden, wird überprüft, ob alle Reagenzien zugeordnet wurden. Dafür wird der berechnete Ähnlichkeitswert zwischen Produkt (p) und Anfragemolekül (a) mit der Anzahl der zugeordneten Linkknoten (l) multipliziert und durch die Zahl der Linkatome (L) geteilt (siehe Formel 5.3).

$$sim(a, p) \cdot \frac{l}{L} \tag{5.3}$$

Für die Zuordnung von Substrukturen des Anfragemoleküls und des Produkts wurde eine Visualisierung geschaffen (siehe auch Abbildung 6.29). Sie ermöglicht es dem Nutzer, verschiedene Zuordnungsmodi zu erkennen und dementsprechend einzuschränken.

5.12.3 Regioselektivität

Bisher war eine Unterscheidung von regioisomeren Substrukturen auf der Ebene des FTree-Deskriptors nicht möglich. Die hier vorgestellte Methode wurde konzipiert, um die Substitutionsmuster an aromatischen Ringen differenzieren zu können. Da aromatische

Ringe planar sind, ist der Ansatz sowohl bei der Verwendung von Datensätzen mit 2D- als auch 3D-Konformationen möglich. Sind keine Atomkoordinaten annotiert, kann theoretisch der Zeichenalgorithmus für 2D-Strukturdiagramme [191] verwendet werden, um 2D-Koordinaten zu generier

Für die Differenzierung der Regioisomere auf Basis des FTree-Deskriptors ist es erforderlich, die FTree-Generierung zu adaptieren. Bisher wurden Atome, die nur einen Bindungspartner besitzen, dem Knoten ihres Bindungspartners zugeordnet. In einem solchen Fall sind zum Beispiel die Moleküle aus Abbildung 5.13 bei einem Feature-Tree-Vergleich nicht differenzierbar. Deshalb werden Schweratome, die nur einen Bindungspartner besitzen, einem eigenen Knoten zugeordnet (siehe Abbildung 5.12).

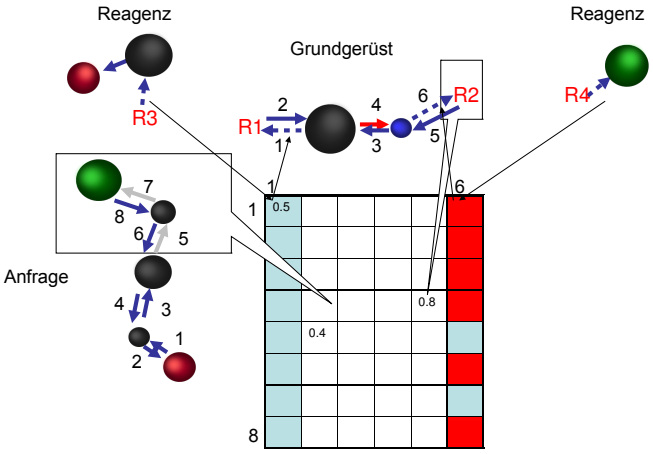
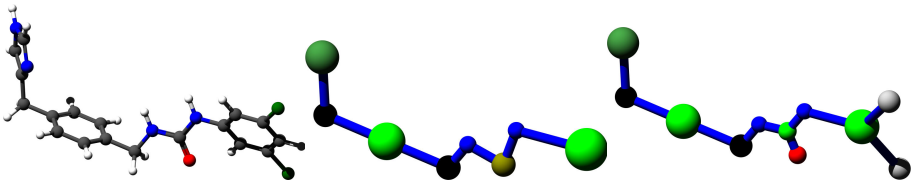


Abbildung 5.11: Eingeschränktes FTree-Matching - Die farbig eingefärbten Zellen werden in der Anfrage/Grundgerüst-Matrix mit den Ähnlichkeitswerten der Vergleiche von Reagenz und korrespondierender Substruktur der Anfrage gefüllt. Ist eine Kombination von Anfrage-FTree-Kante und Reagenz-Linktyp unzulässig, wird ein negativer Wert eingetragen (rote Zellen).



(a) Sch79587 **(b)** **(c)**

Abbildung 5.12: Veränderte FTree-Generierung - Ein Histamin H_3 Rezeptor-Antagonist (Sch79687) [192, 193] (a) und die daraus generierten Feature-Trees unter Verwendung der Standard-Regeln (b) und der veränderten Regeln (c). Hier werden der Carbonylsauerstoff und das Chloratom jeweils einem eigenen Knoten zugeteilt.

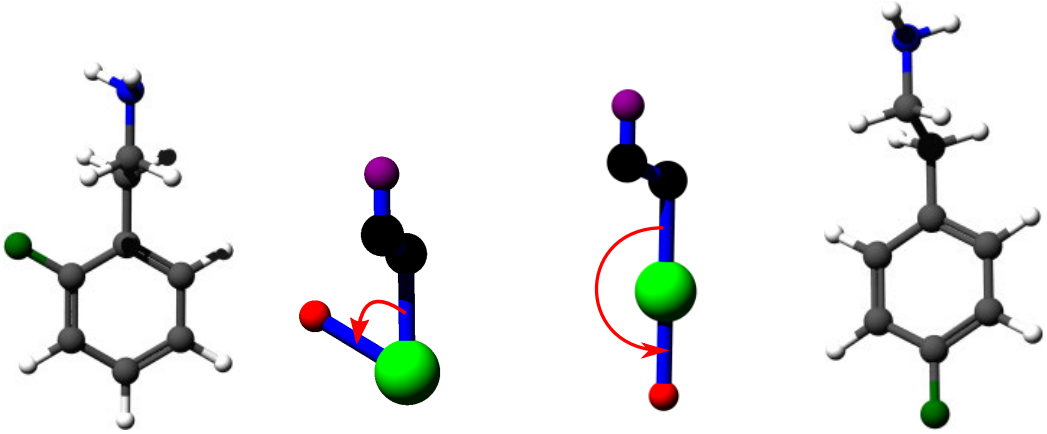


Abbildung 5.13: Berechnung des Regio-Strafterms am Beispiel von 2-(2-Fluoro-phenyl)ethylamin (links) und 2-(4-Fluoro-phenyl)ethylamin (rechts) - Für die Berechnung der Ähnlichkeit der zwei roten Knoten (Fluoratome) werden zunächst die Winkel (rote Pfeile) während des rekursiven Vergleichs berechnet und in die Formel zur Berechnung des Strafterms (siehe Formel 5.4) eingesetzt. Der resultierende Wert wird mit dem Ähnlichkeitswert multipliziert. Die beiden Moleküle (a, b) können dadurch anhand ihrer Feature-Trees unterschieden werden.

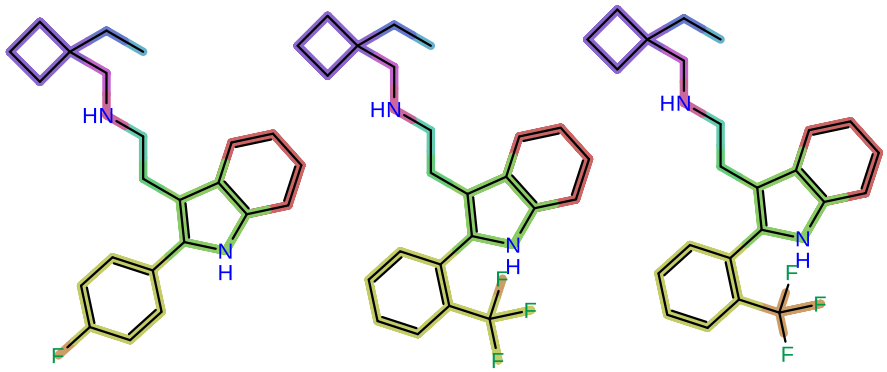


Abbildung 5.14: Einschränkung der Erweiterungsmatches - Um den Strafterm für die Regioselektivität berechnen zu können, sind bei der Zuordnung zweier aromatischer Ringe keine Erweiterungsmatches zulässig. Der Algorithmus ruft sich stattdessen rekursiv für die verbleibenden Teilbäume auf. Die Abbildung soll die Zuordnung der Atome resultierend aus dem FTree-Matching ohne beziehungsweise mit dieser Einschränkung verdeutlichen. Wie bei dem mittleren Molekül eingezeichnet, wird ohne die Einschränkung das Fluoratom des ersten Moleküls nur einem Fluoratom der Trifluor-Methylgruppe zugeordnet. Sind keine Erweiterungsmatches erlaubt, wird die vollständige Trifluor-Methylgruppe zugeordnet, so dass der Strafterm berechnet werden kann. Diese Zuordnung wurde beim rechten Molekül eingezeichnet.

Während eines rekursiven Aufruf des Match-Search-Algorithmus (siehe Kapitel 3.2.3) wird bei der Zuordnung zweier Knoten ein Strafterm für das Zuordnen der Nachfolgeknoten errechnet. Dies geschieht unter Berücksichtigung der bereits zugeordneten Vorgänger- und Vorvorgängerknoten. Für die Errechnung des Strafterms wird zunächst jeweils der Winkel zwischen den Knoten ausgerechnet (siehe Abbildung 5.13). Die Winkel werden in die nachfolgende Gleichung eingesetzt:

$$sim(a,b) \cdot (1 - (\omega \cdot \frac{|\phi_a - \phi_b|}{\pi})) \tag{5.4}$$

Unter Verwendung eines Gewichtungsfaktors ω im Bereich [0,1] wird die absolute Differenz zwischen den beiden Winkeln ϕ_a und ϕ_b durch π geteilt[4]. Der berechnete Strafterm wird nachfolgend mit dem Ähnlichkeitswert der beiden Teilbäume a und b multipliziert. Daraus resultiert, dass unterschiedliche Substitutionsmuster stärker bestraft werden, je größer die darunterliegenden Teilbäume sind, da diese einen größeren Einfluss auf die möglichen 3D-Konformationen der Moleküle besitzen.

Eine weitere Anpassung ist notwendig, da der Match-Search-Algorithmus versucht, im rekursiven Aufruf die gefundenen Matches zu erweitern. Dabei kann es passieren, dass für zwei aromatische Ringe das Match so erweitert wird, dass der Substituent des einen Ringes nur teilweise auf den Substituenten des anderen Ringes gelegt wird (siehe Abbildung 5.14). Wird der Match-Search-Algorithmus regiosensitiv ausgeführt, sind deshalb keine Erweiterungsmatches erlaubt, wenn zwei aromatische Ringe einander zuordnet werden. Für die verbleibenden Teilbäume erfolgt stattdessen immer ein erneuter rekursiver Aufruf.

Generell sind für die Bewertung der Regioselektivität weitere Ansätze denkbar. Statt die Winkelabweichung zu berechnen, kann ein Vergleich der kürzesten Pfade durchgeführt werden. Von Rarey und Stahl [31] wurde dies bereits als alternative Erweiterung des sterischen Knotenprofils eingeführt. Auch wenn für die Berechnung der Winkelabweichung zumindest 2D-Koordinaten berechnet werden müssen, ist der Vorteil des Verfahrens, dass durch die Verwendung eines kontinuierlichen Maßes die wirkliche Geometrie der überlagerten Substrukturen besser repräsentiert wird. Speziell gilt dies bei der Zuordnung von aromatischen Fünf- und Sechsringen. So wird die Zuordnung eines 1,3-substituierten Fünfrings zu einem 1,4-substituierten Sechsring geringer bestraft als

[4]Im Bogenmaß entspricht π der maximalen Differenz von 180°.

die Zuordnung eines 1,3-substituierten Sechsrings. Der kürzeste Pfad ist hingegen in beiden Fällen gleich und geht über zwei Bindungen. Des Weiteren erlaubt die Methode die Unterscheidung der Substitutionsmuster von polyzyklischen Substrukturen, ohne Spezialfälle beachten zu müssen.

5.13 Diversitätsmaße

Nachfolgend sind die Mechanismen, welche zur Diversifizierung der Ergebnisse angewandt werden können, beschrieben.

5.13.1 Diversität in fokussierten Bibliotheken

Werden Bibliotheken generiert, deren Produkte ähnlich zu bestimmten Molekülen sein sollen, sind die gewählten Reagenzien ebenfalls ähnlich zueinander. Dennoch ist es je nach Projektstatus von Vorteil, chemisch unähnlichere Reagenzien zu wählen, so dass ein größeren chemischen Unterraum abgedeckt wird. Um dieses der Ähnlichkeit zur Anfrage gegenläufige Optimierungsziel zu erreichen, bietet *LoFT* mehrere Mechanismen. Zum einen können die Reagenzien mit hierarchisch-agglomerativen Clustering-Verfahren gruppiert werden. Hierfür stehen der Single-Linkage und der Complete-Linkage Clustering-Algorithmus zur Verfügung (siehe auch Kapitel 2.3.3). Zum anderen besteht die Möglichkeit, extern berechnete Cluster-IDs einzulesen (siehe Anhang B.1). Anschließend wird für die Optimierung festgelegt, wie viele Reagenzien aus einem Cluster zugelassen sind. Dabei werden die selektierten Reagenzien für jedes Linkatom des Grundgerüstes separat betrachtet. Wird der erlaubte Schwellenwert überschritten, erfolgt die Bestrafung der jeweiligen Reagenzien abhängig von der Anzahl der Reagenzien aus diesem Cluster. Eine einseitige Wünschbarkeitsfunktion (Formel 5.5) wird verwendet, bei der ω die maximale Bestrafung definiert. Sie liegt im Intervall [0,1] und wird langsam ($n = \frac{1}{2}$), uniform ($n = 1$) oder schnell ($n = 2$) erreicht.

$$cpenalty(x) = \begin{cases} \omega, & \text{wenn } x \geq B \\ \omega \cdot (1 - \frac{(x-A)^n}{(B-A)^n}), & \text{wenn } A < x < B \\ 0, & \text{sonst} \end{cases} \tag{5.5}$$

Neben der Clustereinteilung können ebenso die paarweisen Distanz- beziehungsweise Unähnlichkeitswerte verwendet werden. Als Kriterium dient die durchschnittliche oder minimale paarweise Distanz zwischen den Reagenzien. Auch hier wird eine einseitige

Wünschbarkeitsfunktion verwendet, der Schwellenwert darf allerdings nicht unterschritten werden (Formel 5.6).

$$dpenalty(x) = \begin{cases} \omega, & \text{wenn } x \leq C \\ \omega \cdot (1 - \frac{(D-x)^n}{(D-C)^n}), & \text{wenn } C < x < D \\ 0, & \text{sonst} \end{cases} \tag{5.6}$$

Alle drei Verfahren können parallel angewandt werden. Die Bewertung eines Produktes kann dabei nicht negativ werden (Formel 5.7).

$$pscore(p) = max(0, pscore(p) - cpenalty(p) - dpenalty_{avg}(p) - dpenalty_{min}(p)) \tag{5.7}$$

Um bei der Verwendung großer Fragmenträume nicht die Distanzmatrix berechnen und im Speicher behalten zu müssen, können die Distanzwerte auf Anforderung berechnet werden. Oftmals ist dies effizienter als im Vorfeld die gesamte Matrix zu berechnen. Des Weiteren bietet das integrierte Clustering-Modul die Möglichkeit, ausschließlich Reagenzien mit bestimmten Linktypen beziehungsweise kompatibel zu bestimmten Linktypen zu selektieren. Zusätzlich können die Reagenzien auf ein bestimmtes Eigenschaftsprofil beschränkt werden (siehe Kapitel 5.11).

Um die durchschnittliche und die minimale paarweise Distanz zu berechnen, müssen alle paarweisen Distanzen berechnet werden. Zwischen zwei Reagenzien a und b wird sie berechnet, indem die FTree-Ähnlichkeit sowie die molekularen Eigenschaften in eine Distanzfunktion, angelehnt an die Bewertungsfunktion (siehe Kapitel 5.6), integriert werden:

$$Distanz(a, b) \;=\; (\sum_{i=1}^{|P|} \omega_i \cdot \delta_i(a, b))) \tag{5.8}$$

Dabei gilt, dass sich die Gewichte auf 1,0 aufsummieren: $\sum_{i=1}^{|P|} \omega_i = 1$. Für jede molekulare Eigenschaft p der Menge P wird die absolute Differenz zwischen den Eigenschaftswerten von Reagenz a und b in die jeweilige Wünschbarkeitsfunktion (siehe Abbildung 5.4) eingesetzt: $\delta_p(a, b) = f_p(|p(a) - p(b)|)$ Des Weiteren kann die FTree-Ähnlichkeit zwischen den Reagenzien aufgrund der Problemstellung effizient berechnet werden. Da alle Reagenzien mit dem Grundgerüst verbunden werden, steht die Zuordnung der Linkknoten bereits fest. Zur Berechnung der Unähnlichkeit ($\delta(a, b) = 1 - sim(a, b)$) wird der Match-Search mit den vom Linkknoten ausgehenden FTree-Kanten aufgerufen.

5.13.2 Diversität zwischen fokussierten Bibliotheken

Während einer stochastischen Optimierung wird in jeder Iteration maximal ein Reagenz ausgetauscht. Sollen die n besten fokussierten Bibliotheken gespeichert werden, entstehen dadurch nahezu identische Bibliotheken. Aus diesem Grund kann der Nutzer die minimale Anzahl unterschiedlicher Reagenzien zwischen zwei gespeicherten Bibliotheken spezifizieren. Soll eine neue Bibliothek in die Lösungsliste eingefügt werden, wird sie mit den bisher gespeicherten verglichen. Ist bereits eine Bibliothek mit besserer Bewertung und zu vielen identischen Reagenzien gespeichert, wird die neue Lösung abgelehnt. Anderenfalls wird die neue Lösung akzeptiert und alle anderen Lösungen mit zu vielen identischen Reagenzien werden gelöscht. Der gleiche Ansatz wurde bereits bei FTrees-FS [31] und FlexNovo [34] verwendet. Da die resultierende Lösungsliste reihenfolgeabhängig ist, handelt es sich um einen heuristischen Ansatz.

5.13.3 Diversifizierung der Ergebnisliste beim Cherry-Picking

Analog zu dem in Kapitel 5.13.2 beschriebenen Mechanismus kann beim Cherry-Picking verfahren werden. Um eine möglichst diverse Produktliste zu generieren, wird spezifiziert, wie oft ein Reagenz maximal in den Produkten enthalten sein darf. Analog dazu kann die Anzahl der Reagenzien aus einem Cluster restringiert werden. Eine weitere Möglichkeit ist eine Beschränkung der Ähnlichkeit zweier Reagenzien, beziehungsweise die durchschnittliche Ähnlichkeit der Reagenzien, die Teil der jeweiligen Produkte sind. Dabei werden nur die Reagenzien verglichen, die für die gleiche Substitutionsstelle des Grundgerüstes vorgesehen sind. Die Mechanismen können parallel angewandt werden.

Ein neues Produkt wird in die Lösungsliste eingefügt, wenn es eine bessere Bewertung aufweisen kann als die Produkte, die damit im Konflikt stehen. Von den betroffenen Produkten wird jeweils das Produkt mit der schlechtesten Bewertung aus der Liste gelöscht. Anderenfalls wird das neue Produkt abgelehnt. Die resultierende Produktliste ist dadurch reihenfolgeabhängig.

5.14 3D-Filter

Unter Umständen sind bei der Verwendung des FTree-Deskriptors die entstehenden Produkte zwar auf 2D-Ebene ähnlich der Anfrage, sie lassen sich jedoch im dreidimensionalen Raum nicht mit dem Anfragemolekül überlagern. Dadurch kann es sein, dass

ein Großteil der Produkte der Bibliothek verworfen wird, wenn in einem Nachbearbeitungsschritt ein 3D-Filter wie zum Beispiel FlexS [194, 195] oder ROCS [196, 197] angewandt wird. Es ist jedoch in Bezug auf die Laufzeit des Programms zu aufwendig, einen 3D-Filter während der Optimierung einzusetzen. Folglich wurde eine Möglichkeit gesucht, 3D-Filter sowohl in einem Vor- als auch in einem Nachbearbeitungsschritt verwenden zu können. Um einen ineinandergreifenden Arbeitsablauf anbieten zu können, wurde eine FlexS-Anbindung geschaffen (siehe auch Anhang B.2). Dabei verwendet FlexS die FTree-Matchings zwischen zwei Molekülen für die schnelle und automatische 3D-Überlagerung der Strukturen. Die FTree-Matchings fungieren dabei als Startpositionierung der Substrukturen, um die Orientierung und Konformation des einen Moleküls in Bezug auf das andere vorherzusagen. Durch die initiale Positionierung ist es möglich die Berechnung zu beschleunigen, beziehungsweise eine Aussage über das FTree-Matching zu treffen. In *LoFT* findet die Anbindung an FlexS an zwei Stellen Anwendung. Zum einen als Produktfilter, der durch das Ausgabeformat mühelos in den Arbeitsablauf des Benutzers eingebunden werden kann. Zum anderen um ungeeignete Reagenzien vor der Optimierung herauszufiltern. Dafür werden alle Reagenzien, die den korrekten Linktyp besitzen und den verwendeten Eigenschaftsfilter passieren, mit dem ausgewählten Grundgerüst verknüpft. Das entstandene Fragment wird mit der Anfragestruktur überlagert. Durch die Verknüpfung mit dem Grundgerüst kann die Lage der Substituenten besser abgeschätzt werden. Anschließend erfolgt die Filterung der Reagenzien anhand ihrer Bewertung durch FlexS.

6 Resultate und Diskussion

Zunächst wird evaluiert, wie sich die Verwendung der *NAOMI*-Bibliothek im Kontext von *LoFT* auswirkt. Anschließend wird das Verfahren exemplarisch anhand unterschiedlicher Fallstudien und Designszenarien evaluiert. Abschließend wird die Performance von *LoFT* bei der Generierung von Bibliotheken unterschiedlicher Größe betrachtet. Alle Berechnungen wurden auf einem Intel Core 2 Duo 3 GHz mit 4 GB RAM und openSUSE 11.1 [198] durchgeführt.

6.1 Auswirkungen der *NAOMI*-Bibliothek

Nachfolgend wird der Einfluss der *NAOMI*-Bibliothek auf die Verwendung von Fragmenträumen und des Feature-Tree-Deskriptors untersucht. Hierfür werden die auf der Flex*- und der *NAOMI*-Bibliothek basierenden *LoFT*-Versionen (nachfolgend *LoFT* 1.0 beziehungsweise 2.0 genannt) verglichen.

6.1.1 Einlesen von Fragmenträumen

In diesem Abschnitt werden Laufzeit und Speicherbedarf beim Einlesen der in den Fallbeispielen (siehe Kapitel 6.2) verwendeten Fragmenträume untersucht. Zusätzlich wird eine CXCR3-Bibliothek[199] betrachtet[1]. Sie war ursprünglich zur Validierung des ersten Prototypen von *LoFT* [32] gedacht. Zum damaligen Zeitpunkt war jedoch die Speicheranforderungen zu hoch. Um die Untersuchung nicht auf Fragmenträume, die

[1]Der CXC-Motiv-Chemokinrezeptor 3 (CXCR3) spielt bei der Entstehung von entzündlichen Krankheiten eine Rolle [200]

eine einzelne kombinatorische Bibliothek enthalten, zu beschränken, wird der frei verfügbare BRICS-Fragmentraum [163] verwendet. Der Fragmentraum wurde aus aktiven Molekülen des World Drug Index (WDI) [161] und einer Auswahl von Molekülen mit wirkstoffähnlichen Eigenschaften des ZINC-Datensatzes (*Drug-like subset*) [201] durch Anwendung retrosynthetischer Schnittregeln generiert.

Eine Auflistung der Größe der Fragmenträume findet sich in Tabelle 6.1.

Datensatz	*LoFT* 1.0/FlexNovo	*LoFT* 2.0
BRICS	22 343	22 339
H3	10 315	10 315
CDK2-1	11 266	11 266
CDK2-2	29 653	29 640
5HT2a	25 568	25 558
CXCR3	65 607	65 560

Tabelle 6.1: Anzahl der eingelesenen Fragmente - Bei *LoFT* 2.0 sind es weniger Fragmente, da einige Fragmente fehlerhaft sind und deshalb abgelehnt werden.

Unter Verwendung der *NAOMI*-Bibliothek erfolgt die Initialisierung der Fragmente bereits beim Einlesen des Fragmentraumes. Dies bedeutet, dass die Fragmente bereits chemisch validiert und annotiert sind. Das ist bei *LoFT* 1.0 – beziehungsweise FlexNovo [34], dessen Fragmentraummodul verwendet wird – nicht der Fall. Die Fragmente werden entweder mit dem Menü-Kommando **INIT** oder intern vor Berechnung der Deskriptoren initialisiert.

Datensatz	*LoFT* 1.0/FlexNovo	*LoFT* 1.0 (initialisiert)	*LoFT* 2.0
BRICS	0:07 min	1:15 min	0:05 min
H3	0:04 min	0:49 min	0:02 min
CDK2-1	0:04 min	0:57 min	0:03 min
CDK2-2	0:12 min	3:20 min	0:08 min
5HT2a	0:11 min	2:55 min	0:08 min
CXCR3	1:04 min	7:14 min	0:19 min

Tabelle 6.2: Laufzeiten beim Einlesen von Fragmenträumen in Minuten und Sekunden

Tabelle 6.2 zeigt die Laufzeiten für das Einlesen und Initialisieren der Fragmente.

Bei den kombinatorischen Bibliotheken ergibt sich eine deutliche Laufzeitreduzierung um Faktor 22-25. Da für den CXCR3-Raum bei *LoFT* 1.0 virtueller Speicher verwendet wird, benötigt das Einlesen und die Initialisierung des CXCR3-Raumes auf dem verwendeten Rechner inklusive der Systemzeit 13 Minuten und 38 Sekunden. Zudem zeigt sich, dass durch *NAOMI* die vollständige Initialisierung bereits schneller ist, als das Einlesen der im MOL2-Format [175] annotierten Daten mit dem Fragmentraum-Modul von FlexNovo.

Bei *LoFT* 2.0 werden die Fragmente parallel initialisiert. Aufgrund der benötigten Dateioperationen skaliert die Laufzeit jedoch nicht linear mit der Anzahl der Prozessoren[1]. Es zeigt jedoch, dass die *NAOMI*-Bibliothek die Parallelisierung anderer rechenintensiver Algorithmen auf Molekülen erlaubt.

Datensatz	*LoFT* 1.0/FlexNovo	*LoFT* 2.0
BRICS	1,2 GB	0,4 GB
H3	0,7 GB	0,3 GB
CDK2-1	0,84 GB	0,3 GB
CDK2-2	1,4 GB	0,5 GB
5HT2a	1,2 GB	0,5 GB
CXCR3	4,7 GB	0,9 GB

Tabelle 6.3: Speicherbedarf nach dem Einlesen und Initialisieren des jeweiligen Fragmentraumes - Die Werte sind gerundet in Gigabyte (GB) angegeben.

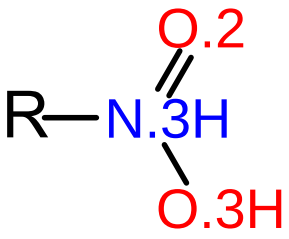


Abbildung 6.1: Fehlerhafte Nitrogruppe - Sowohl Stickstoff als auch der einfach gebundene Sauerstoff sind fälschlicherweise protoniert. Der Stickstoff ist dadurch fünfbinding. Die Darstellung zeigt die korrespondierenden Sybylatomtypen des MOL2-Formates [175].

Der Speicherbedarf ist für die Beispielräume mehr als halbiert (siehe Tabelle 6.3). Der CXCR3-Raum ist der größte der verwendeten Fragmenträume und die enthaltenen Fragmente besitzen im Durchschnitt die meisten Schweratome. Durch die Nutzung der

[1]Die Untersuchungen im Kontext der *NAOMI*-Veröffentlichung [35] zeigten, dass bei der Dateiformat-Konvertierung von Molekülen auf einem Rechner mit zwei 2,53 GHz Intel Xeon Prozessoren die Laufzeit durch die Parallelisierung um Faktor 1,4 verbessert wird.

NAOMI-Bibliothek wird der Speicherbedarf auf ein Fünftel reduziert. Dadurch wird die Verwendung dieses Raumes erst praktikabel. Denn auf 32Bit-Systemen konnte der CXCR3-Raum bisher nicht initialisiert werden, da der Speicherbedarf auf über drei Gigabyte steigt. Tabelle 6.1 zeigt zudem, dass bei *LoFT* 2.0 weniger Fragmente geladen werden. Dies liegt daran, dass die Eingabedateien einige Fragmente enthalten, die chemisch nicht valide sind. Die meisten Fragmente können jedoch korrigiert werden. Dabei handelt es sich hauptsächlich um Fragmente mit fehlerhaften Nitrogruppen (siehe Abbildung 6.1). Des Weiteren werden falsche Verknüpfungsregeln, wie beispielsweise das Terminieren eines Links mittels Doppelbindung und Wasserstoff, durch die chemische Validierung des Fragmentraumes erkannt und abgelehnt.

Insofern zeigt sich, dass die *NAOMI*-Bibliothek nicht nur einen positiven Einfluss auf die Geschwindigkeit und den Speicherbedarf von *LoFT* hat, sondern vielmehr notwendig ist, um die Arbeit mit Fragmenträumen im Kontext kombinatorischer Bibliotheken zu erlauben.

6.1.2 Generierung von Feature-Trees

Um eine konsistente Menge von Deskriptoren verwenden zu können, werden die Feature-Trees unter Zuhilfenahme des neuen Chemie-Modells generiert. Insofern ist zu prüfen, inwieweit sich die Änderungen bei der Generierung der Feature-Trees und somit beim paarweisen Vergleich auswirken. Dafür werden zwei gängige Datensätze verwendet:

- Der DUD-Datensatz [202] (*Database of Useful Decoys*) enthält für vierzig pharmakologisch relevante Proteine aktive Moleküle mit bekannter Bindungsaktivität. Für jedes aktive Molekül existiert eine Menge von *Decoys*. Diese inaktiven Moleküle besitzen ähnliche physikochemische Parameter, jedoch eine unterschiedliche Topologie. Tabelle A.2 (Anhang A.2) listet die Anzahl und Einteilung der Moleküle auf.

- Von Hert et al. [203] wurde ein Datensatz mit elf Aktivenklassen zusammengestellt. Insgesamt besteht der Datensatz aus 102 535 Molekülen der MDDR-Datenbank [204] (*MDL Drug Data Report*). Die Anzahl der Moleküle der jeweiligen Klasse ist in Tabelle A.1 (Anhang A.1) gelistet. Allerdings sind 1183 Angiotensin II Inhibitoren nicht richtig zugeordnet, so dass eine überarbeitete

Einteilung verwendet wurde [166]. Zudem ist zu beachten, dass es einige Substanzen gibt, welche zwei oder drei Aktivenklassen zugeordnet wurden oder nicht mehr im MDDR-Datensatz von 2008 enthalten sind.

Um den Einfluss der neuen Feature-Tree-Generierung genau spezifizieren zu können, wurden die Feature-Trees auf unterschiedliche Arten generiert (siehe Tabelle 6.4).

Modus	Beschreibung
FTrees	Mit FTrees unter Verwendung der Standard-Initialisierungsparameter.
FTrees NAOMI	Mit FTrees unter Verwendung der neuen Module zum Einlesen und Initialisieren der Moleküle (*NAOMI*). Des Weiteren wurde eine Funktion zum Konvertieren in die alte Molekülstruktur verwendet. Anschließend läuft die normale Generierung der Feature-Trees.
NAOMI FTMode	Die Feature-Trees werden generiert, wie es in Kapitel 4.4 beschrieben wurde. Wie bei FTrees werden terminale Schweratome dem Knoten des Bindungspartners zugeordnet.
NAOMI	Die Feature-Trees werden generiert, wie es in Kapitel 4.4 beschrieben wurde. Terminale Schweratome bekommen einen eigenen Knoten.

Tabelle 6.4: Verschiedene Modi zum Generieren der Feature-Trees. Zur besseren Vergleichbarkeit werden auch unter Verwendung von *NAOMI* die Standard-Protonierungszustände zugewiesen.

Datensatz	FTrees	FTrees *NAOMI*	*NAOMI* FTMode	*NAOMI*
Hert	23:16 min	4:10 min	1:10 min	1:17 min
DUD	27:48 min	5:14 min	1:30 min	1:41 min

Tabelle 6.5: Laufzeiten bei der Konvertierung ins FTrees-Datenformat in Minuten und Sekunden.

Die Datensätze werden mit FTrees beziehungsweise dem *NAOMI*-Konverter eingelesen, die Feature-Tree-Deskriptoren werden generiert und im FTrees-Datenformat wieder herausgeschrieben. Tabelle 6.5 zeigt die Laufzeiten bei der Konvertierung für beide Datensätze. Im Vergleich zur Konvertierung mit FTrees ist die Konvertierung ungefähr fünfeinhalb Mal schneller, wenn die *NAOMI*-Initialisierung der FTree-Generierung vorgeschaltet wird. Erfolgt die Konvertierung mit dem *NAOMI*-Konverter, läuft der Vorgang ungefähr siebzehn Mal schneller ab. Die kürzeren Laufzeiten bei der Generierung

mit dem FTMode resultieren aus der kompakteren Repräsentation der Feature-Trees (160 MB statt 200 MB Ausgabedaten beim DUD-Datensatz).

Für *LoFT* bedeuten die verkürzten Laufzeiten, dass die Feature-Trees nicht mehr in externen Dateien gespeichert werden müssen. Stattdessen können sie vor einer Optimierung, bei der die Feature-Tree-Ähnlichkeit in der Ziel- oder Sortierfunktion verwendet wird, erstellt werden. Dadurch werden Inkonsistenzen vermieden und der Nutzer muss sich nicht um die Aktualisierung der Daten kümmern.

Bedeutender als die Performance der Generierung ist jedoch, dass die neu generierten Feature-Trees zumindest eine gleich bleibende Qualität in Bezug auf den Ähnlichkeitsvergleich aufweisen. Dafür wurden sogenannte Anreicherungsexperimente[1] mit dem Hert- und dem DUD-Datensatz durchgeführt. Die Feature-Trees selbst wurden auf unterschiedliche Weise generiert (siehe Tabelle 6.4) und die Vergleiche mit der aktuellen FTrees Version 2.3.0 [205] durchgeführt. Zusätzlich wurde ein Anreicherungsexperiment gemacht, bei dem der Strafterm für die Regioselektivität mit 0.2 gewichtet wurde (*NAOMI* Regio2). Die Standard-Parameter des Match-Search-Algorithmus wurden nicht verändert.

Abbildung 6.2 zeigt das Anreicherungsdiagramm für den Hert-Datensatz, Abbildung 6.3 für den DUD-Datensatz. Die Werte sind gemittelt über die Werte der einzelnen Aktivitätsklassen. Dabei führt die Generierung mit *NAOMI*, bei der die terminalen Atome dem Bindungspartner zugeordnet werden, zu den besten Ergebnissen. Generell führen die verschiedenen Verfahren in etwa zu gleichen Anreicherungswerten. Dies ist zu erwarten, da weder der Feature-Tree-Deskriptor noch der Vergleichsalgorithmus substantiell verändert wurden. Vielmehr war das Ziel, dass der Feature-Tree-Deskriptor konsistent mit den anderen verwendeten Deskriptoren generiert wird (vergleiche Kapitel 4.4). Werden die Anreicherungen der einzelnen Aktivitätsklassen gesondert betrachtet, sind jedoch signifikante Unterschiede festzustellen. Anhang A.1 zeigt die Diagramme für die einzelnen Aktivitätsklassen des Hert-Datensatzes und Anhang A.2 die Diagramme für die Aktivitätsklassen des DUD-Datensatzes. Wie zu erwarten, ist bei einigen Klassen die Verwendung der Regioselektivität von Vorteil. Ein Beispiel hierfür sind die HIV-1 Protease-Inhibitoren des Hert-Datensatzes. Umgekehrt, wie bei FGFR1 des

[1] Wird unter Verwendung eines aktiven Moleküls die Datenbank durchsucht, entsteht durch die Berechnung der Ähnlichkeitswerte eine Ordnung. Ein Ähnlichkeitsmaß ist umso besser, je mehr aktive Moleküle einen hohen Ähnlichkeitswert erhalten, sich also anreichern.

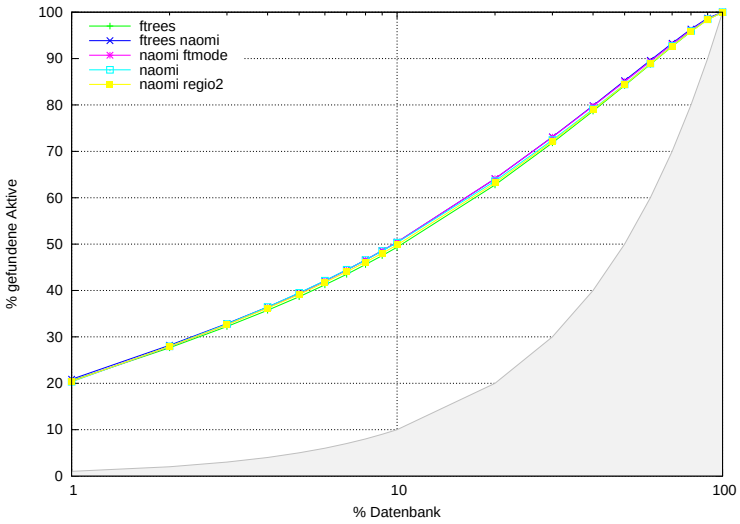


Abbildung 6.2: Die Anreicherungskurven gemittelt über alle Aktivitätsklassen des Hert-Datensatzes (Anhang A.1).

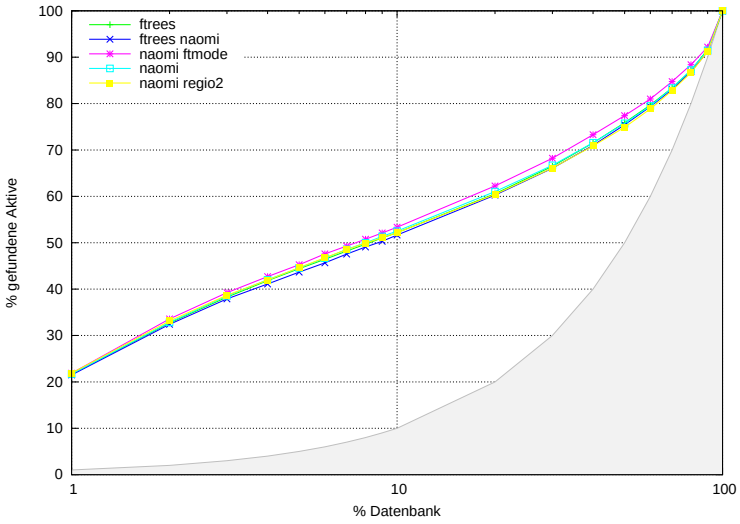


Abbildung 6.3: Die Anreicherungskurven gemittelt über alle Aktivitätsklassen des DUD-Datensatzes (Anhang A.2).

DUD-Datensatzes, ist die Betrachtung der Regioselektivität von Nachteil, wenn Aktive mit unterschiedlichen Substitutionsmustern existieren. Beachtenswert ist auch, dass sich die Zuordnung der terminalen Atome stark auf die Anreicherung auswirkt. Dies führt bei einigen Datensätzen zur Verbesserung, bei anderen zur Verschlechterung der Anreicherung. Da der Match-Search-Algorithmus nur eine limitierte Anzahl initialer Schnitte und Erweiterungsmatches macht (vergleiche Kapitel 3.2.3), ändert sich durch die zusätzlichen Knoten das Matching. Außerdem kann es sein, dass in dem neuen Modus andere Teilstrukturen in die Erweiterungsmatches eingebunden werden (siehe auch Fallbeispiel 6.2.3.2). Um dies zu vermeiden, kann die Gewichtung der Nullmatches erhöht werden. Die Evaluierung der unterschiedliche Parametrisierungen ist im Rahmen der vorliegenden Arbeit jedoch nicht möglich.

Interessant sind die Aktivitätsklassen, bei denen FTrees bessere Ergebnisse liefert. Die Untersuchung dieser Ergebnisse resultierte bereits in einige Korrekturen und Verbesserungen bei der Generierung durch *NAOMI*. Da sich die Generierung der Standardprotonierungszustände derzeit noch in der Entwicklungsphase befindet, ist eine weitere Verbesserung der Ergebnisse von *NAOMI* zu erwarten.

6.2 Fallstudien

Der Knowledge Space [206] ist ein frei verfügbarer Fragmentraum, der insgesamt 82 kombinatorische Syntheseprotokolle umfasst. Die Kombination der 10 879 Fragmente ergibt über zwölf Milliarden virtuelle Produkte, die synthetisiert werden können. Aus den 82 Protokollen wurden vier ausgewählt, um Fragmenträume zu entwerfen, mit denen die Funktionalität von *LoFT* validiert werden kann. Die Reagenzien der jeweiligen Bibliothek wurden entsprechenden Anbieterkatalogen entnommen. Es handelt sich zum Beispiel um Standard-Amine, Aniline und Carbonsäuren.

Aus diesen Bibliotheken sollen fokussierte Bibliotheken entworfen werden, deren Produkte ähnlich zu biologisch aktiven Molekülen aus der Literatur sind. Im Folgenden wird untersucht, wie die entwickelten Mechanismen ineinander greifen. Dafür werden iterativ weitere Kriterien bei der Optimierung hinzugefügt, bis die Subbibliotheken individuellen Designkriterien entsprechen (siehe Abbildung 6.4). Um eine diverse Reagenzienauswahl zu erhalten, wird ein Reagenz aus jedem Ähnlichkeitscluster erlaubt. Die Reagenzien wurden anhand ihrer Feature-Tree-Ähnlichkeit mit dem Complete-Linkage

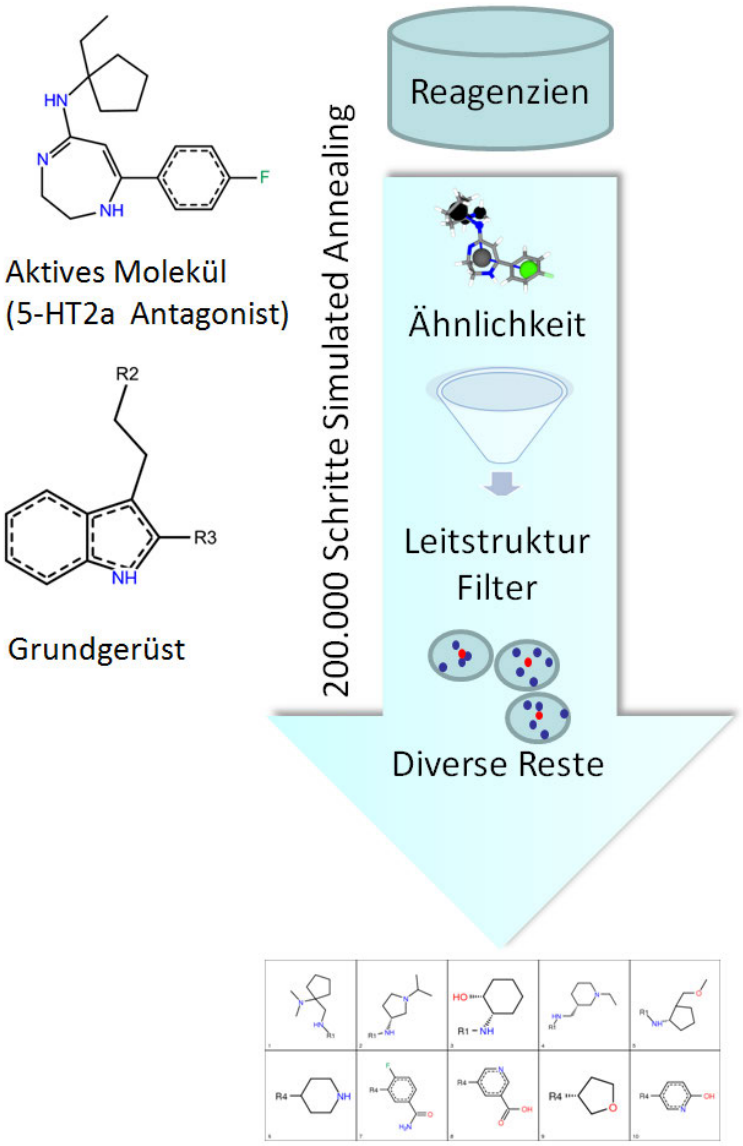


Abbildung 6.4: Iterativer Designprozess - Iterativ werden weitere Kriterien hinzugefügt, bis die Bibliothek dem Designziel entspricht. In diesem Beispiel wird zunächst eine Bibliothek auf Ähnlichkeit zu einem Anfragemolekül hin optimiert. Um Strukturen zu erhalten, die zudem den Kriterien von Leitstrukturen entsprechen, wird der Oprea-Filter [104] angewendet. Um chemisch unterschiedliche Reagenzien auszuwählen, wird die Anzahl der Reagenzien aus einem Cluster beschränkt.

Algorithmus in Cluster eingeteilt. Der Distanzschwellenwert lag bei 0,1. Zudem wurden nur die chemischen Knoteneigenschaften betrachtet. Dies führt empirisch zu einer besseren Clustereinteilung. Um Produkte zu erhalten, die sich als Leitstrukturen eignen, wird in den Fallstudien, in Anlehnung an die Kriterien von Oprea [104], der nachfolgend vorgestellte Produktfilter verwendet. Werden die Eigenschaften stattdessen in die Bewertungsfunktion eingebunden, beschreiben diese Eigenschaftswerte die Punkte B und C der Wünschbarkeitsfunktion (siehe Abbildung 5.4), beziehungsweise B ist 0. Die in Klammern angegeben Werte stellen dementsprechend den Grenzwert A beziehungsweise D dar, bei denen eine Bewertung mit 0 erfolgt. Die Kriterien sind:

- Molekulargewicht $\leq$ 450 (600)
- Anzahl der Ringe $\leq$ 4 (6)
- (-6) -3,5 $\leq$ clogP $\leq$ 4,5 (7)
- Anzahl der Wasserstoffbrückendonoren $\leq$ 5 (8)
- Anzahl der Wasserstoffbrückenakzeptoren $\leq$ 8 (12)
- Anzahl der rotierbaren Bindungen $\leq$ 10 (15)

Die Gewichtung des Regioselektivitätsstrafterms wird auf 1 gesetzt, wenn bei der Optimierung eine Unterscheidung der Substitutionsmuster erfolgen soll. Aus diesem Grund werden die Feature-Trees für die Fallbeispiele so generiert, dass terminale Schweratome einen eigenen Knoten erhalten. Schließlich kann das Matching eingeschränkt werden, so dass bestimmte Kanten des Anfrage-FTrees lediglich bestimmten Reagenz-Linkkanten zugeordnet werden können. Dies gewährleistet die Zuordnung von Grundgerüst und einer bestimmten Substruktur des jeweiligen Anfragemoleküls.

Um im Rahmen dieser Arbeit einen Vergleich zu ermöglichen, werden Subbibliotheken mit jeweils fünf Reagenzien bei zwei Substitutionsstellen (5x5 Bibliothek), beziehungsweise jeweils vier Reagenzien bei drei Substitutionsstellen des jeweiligen Grundgerüstes generiert (4x4x4 Bibliothek).

6.2.1 Parametrisierung

Um bei der Optimierung gute Resultate zu erzielen, ist eine an das Problem angepasste Parametrisierung der Optimierungsalgorithmen Voraussetzung. Die Schwierigkeit besteht darin, dass die Bewertungsfunktion nicht a priori bekannt ist. Sie kann sich drastisch von Szenario zu Szenario anhand der Gewichtung und Art der Kriterien unterscheiden [57]. Ebenso spielen weitere Faktoren, wie zum Beispiel die Anzahl der verfügbaren Reagenzien und die Größe der zu generierenden Subbibliothek, eine Rolle. Auch die Verwendung der Eigenschaftsfilter und der Strafterme für Diversität hat einen nicht zu unterschätzenden Einfluss. Ein weiterer Faktor ist die Vorsortierung der Reagenzien. Durch die Sortierung kann eine gute Ausgangsbibliothek generiert werden, so dass weniger Schritte bei der Optimierung notwendig sind, bis der Algorithmus konvergiert.

Bei der Simulierten Abkühlung führen generell Abkühlungsverläufe, bei denen länger mit einer niedrigen Temperatur optimiert wird, zu besseren Ergebnissen. Dennoch ist es ebenso wichtig, dass ausreichend Schritte bei hoher Temperatur erfolgen, um lokale Maxima verlassen zu können [57].

Abbildung 6.5 zeigt den Abkühlungsprozess für eine exponentielle Abkühlung mit einer Starttemperatur von 1,0 bei der Simulierten Abkühlung über 200 000 Schritte für unterschiedliche Abkühlungsfaktoren. Die Wahrscheinlichkeit für die Annahme einer schlechter bewerteten Bibliothek ergibt sich durch Einsetzen der Bewertungsdifferenz ΔE und der Temperatur T zum Zeitpunkt t in die Formel $e^{(-\frac{\Delta E}{T_t})}$. Exemplarisch zeigt Abbildung 6.6 die Wahrscheinlichkeiten für die Annahme einer um 0,05; sowie 0,1; 0,2 und 0,3 schlechter bewerteten Bibliothek in Abhängigkeit von der jeweiligen Temperatur.

Die Untersuchungen für die Fallbeispiele der ersten Veröffentlichung [32] ergaben, dass bei 20 000 bis 40 000 Reagenzien über die unterschiedlichen Szenarien hinweg die Verwendung einer Schrittweite von 200 000 Schritten, einer Starttemperatur von 1,0 und eines exponentiellen Abkühlungsfaktors von 0,99992 zu guten Ergebnissen führt. In diesem Fall wird statistisch gesehen fünf- bis zehnmal versucht, ein bestimmtes Reagenz der fokussierten Bibliothek hinzuzufügen. Aus diesem Grund wird für die hier gezeigten Fallbeispiele eine Simulierte Abkühlung mit den genannten Parametern angewandt. Erfolgt mehr als 50 000 Schritte kein Austausch der Reagenzien, bricht der Algorithmus

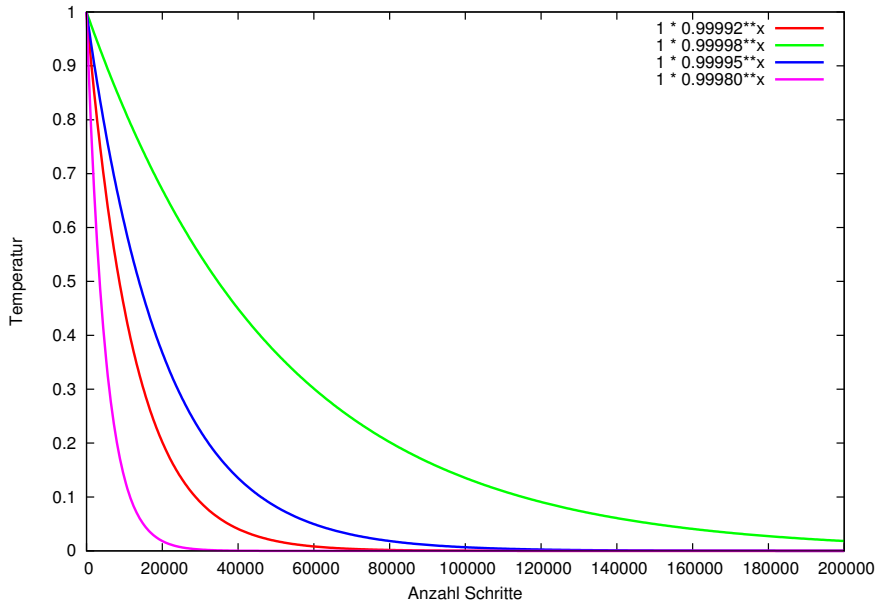


Abbildung 6.5: Temperaturverhalten bei der Simulierten Abkühlung unter Verwendung unterschiedlicher Abkühlungsfaktoren - Die Starttemperatur beträgt 1,0.

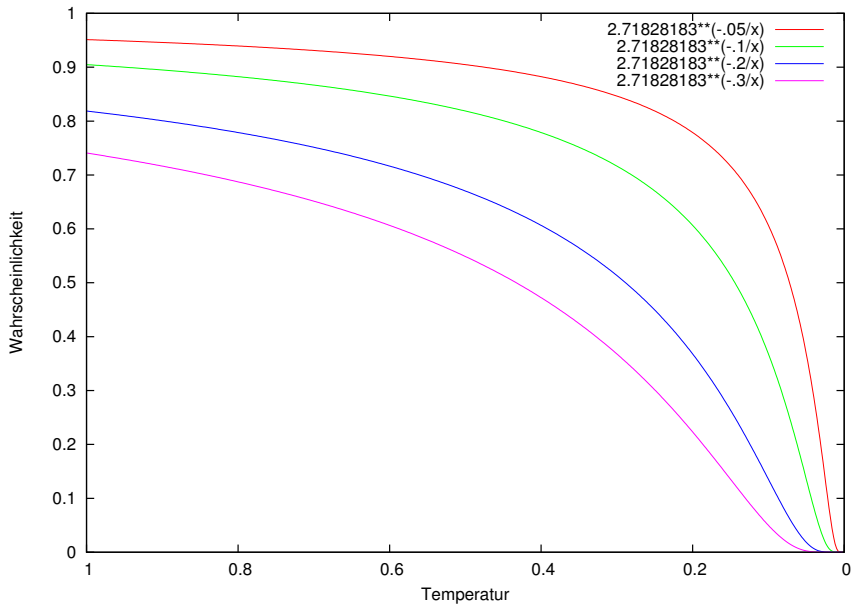
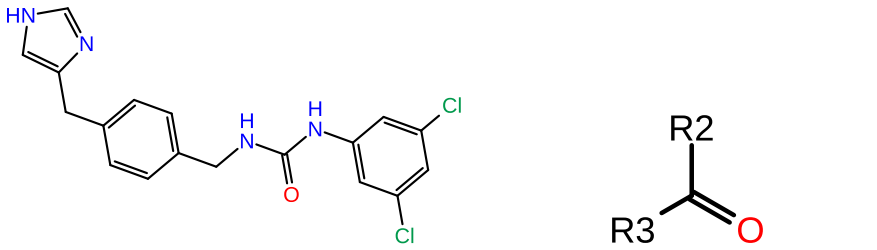


Abbildung 6.6: Wahrscheinlichkeit für die Annahme einer schlechter bewerteten Bibliothek in Abhängigkeit von der Temperatur. - Exemplarisch werden die Graphen für eine Verschlechterung der Bibliotheksbewertung um 0,05; 0,1; 0,2 und 0,3 dargestellt.

vorzeitig ab. Falls nicht anders angegeben, beträgt der Initialwert (*Seed*) des Pseudozufallszahlengenerators 1. Die verwendeten Parameter werden bei jeder Optimierung in der Ausgabe-Datei von *LoFT* dokumentiert (siehe Anhang B.2).

6.2.2 Histamin-H_3-Rezeptor

Der Histamin-H_3-Rezeptor ist ein G-Protein-gekoppelter Rezeptor (GPCR) und befindet sich in der Zellmembran. Er steuert die Ausschüttung von Histamin [207] und anderen Neurotransmittern wie Serotonin und Acetylcholin [193]. Dadurch ist der H_3-Rezeptor an der Steuerung von Blutdruck und Körpertemperatur, sowie von Hunger- und Durstgefühl beteiligt. H_3-Rezeptor Antagonisten werden entwickelt, um verschiedenste neurologische und kognitive Störungen wie zum Beispiel Epilepsie zu therapieren [208]. In diesem Fallbeispiel wird für die Exploration des chemischen Raumes um einen bekannten H_3-Rezeptor-Antagonisten (Sch79687, siehe Abbildung 6.7a) [192], eine Harnstoffbibliothek mit einer Carbonylgruppe als Grundgerüst [209] (siehe Abbildung 6.7b) und 10 314 Reagenzien verwendet. Alle Reagenzien sind zu beiden Linkatomen R2 und R3 des Grundgerüstes kompatibel.



(a) H_3-Antagonist Sch79687 **(b)** H_3 Grundgerüst

Abbildung 6.7: Anfrage (a) und Grundgerüst (b) für die H_3-Rezeptor Bibliothek

Zunächst wird eine Simulierte Abkühlung mit 200 000 Schritten durchgeführt und die Ähnlichkeit zum Anfragemolekül Sch79687 (siehe Abbildung 6.7a) als alleiniges Kriterium verwendet. Die daraus resultierende 5x5 Bibliothek (siehe Abbildung 6.8) enthält für das R3-Linkatom des Grundgerüstes Dichloro-Aniline in allen Variationen. Die selektierten Reagenzien des R2-Linkatoms zeigen ebenfalls unterschiedliche Substitutionsmuster des Phenylrings. Dies führt zu unterschiedlichen Geometrien der resultierenden Produkte.

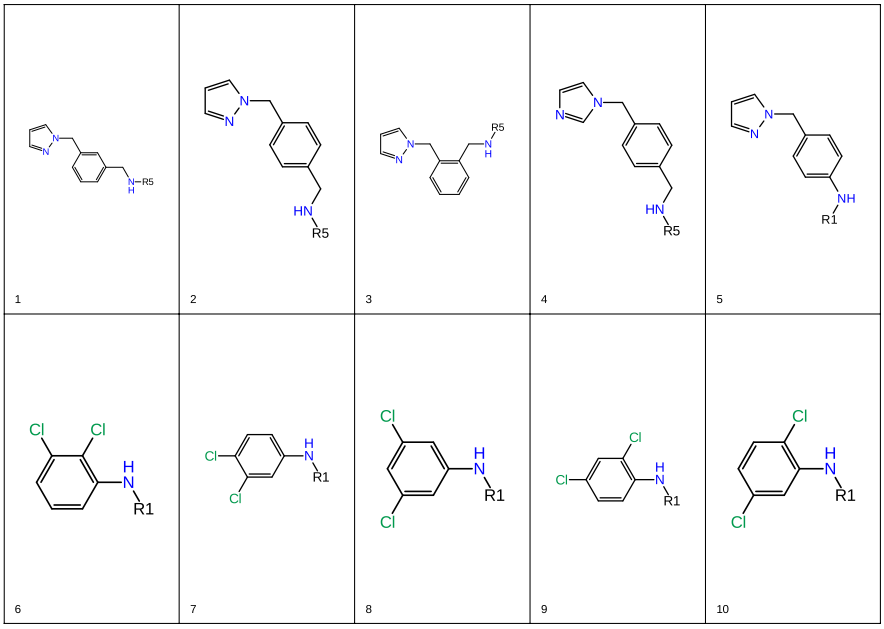


Abbildung 6.8: Bibliothek H_3-1 - Resultierende Reagenzienauswahl (5x5 Bibliothek) in Bezug auf die Ähnlichkeit zum Anfragemolekül Sch79687.

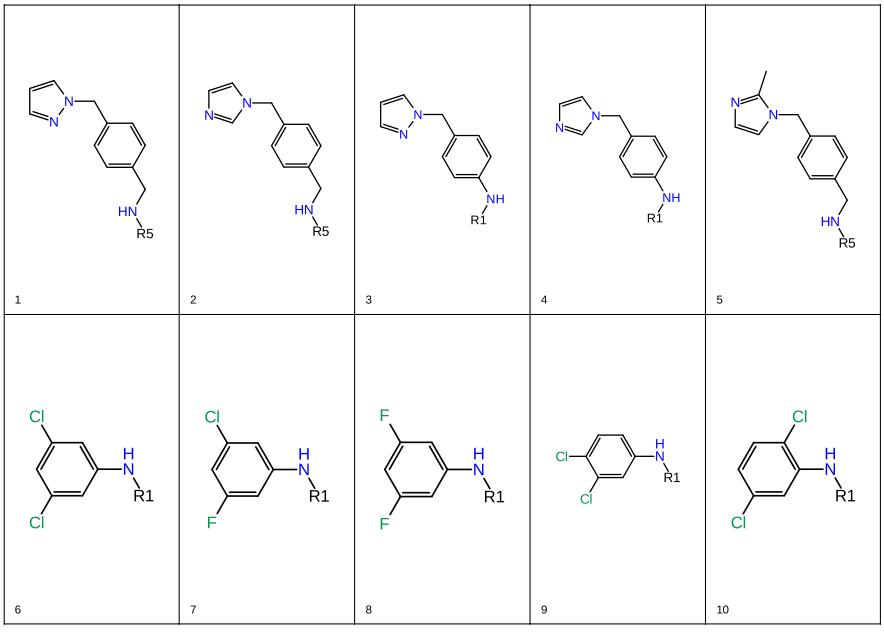


Abbildung 6.9: Bibliothek H_3-2 - 5x5 Bibliothek optimiert auf die Ähnlichkeit zum Anfragemolekül unter Berücksichtigung der Substitutionsmuster. Dafür wurde die Regioselektivität mit 1.0 gewichtet.

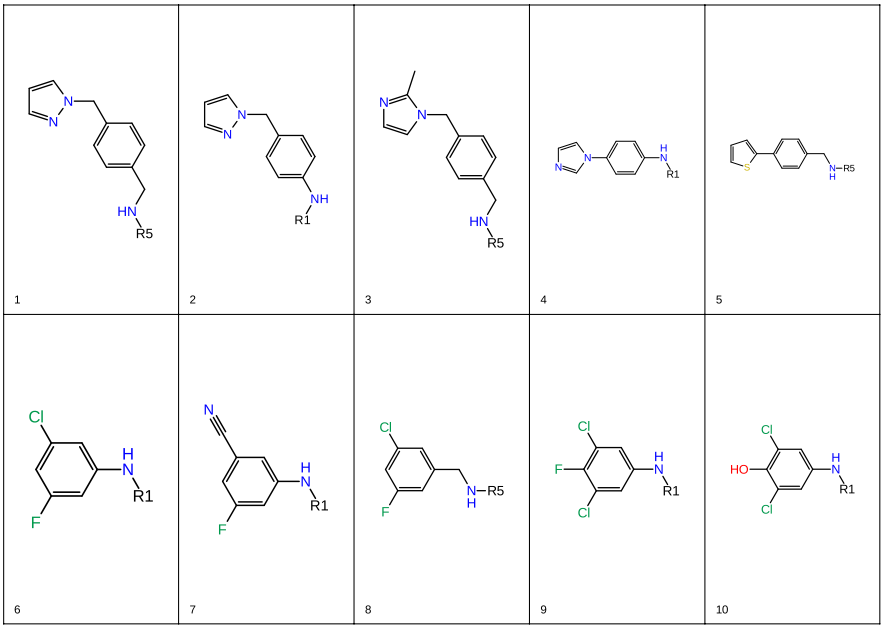


Abbildung 6.10: Subbibliothek H_3-3 - Die 5x5 Bibliothek enthält lediglich ein Reagenz aus jedem Cluster. Die gewählten Reagenzien sind untereinander unähnlicher, dennoch tendieren sie zu den gewünschten Substitutionsmustern.

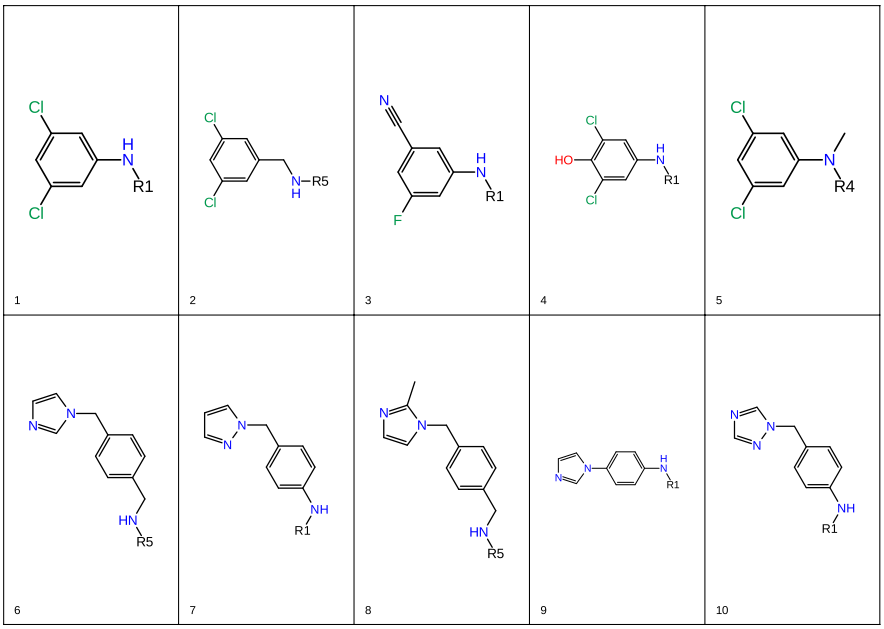


Abbildung 6.11: Subbibliothek H_3-4 - Die 5x5 Bibliothek wurde auf Ähnlichkeit zur Anfrage optimiert und enthält nur ein Reagenz aus jedem Cluster. Die Regioselektivität wurde mit 1.0 gewichtet und die Produkte entsprechen dem Oprea-Filter.

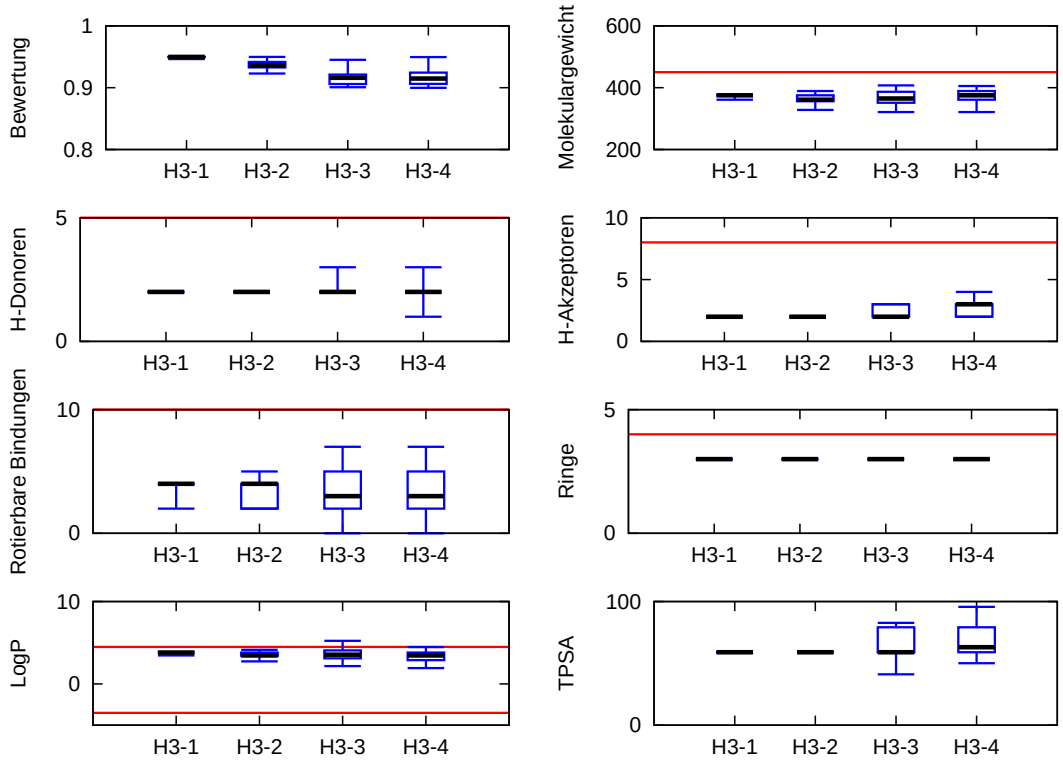


Abbildung 6.12: Eigenschaftsprofile der generierten H_3-Subbibliotheken - Die Abbildung zeigt Box-Whisker-Plots für die bedeutendsten Eigenschaften. Die Grenzwerte des Oprea-Filters sind mittels roter Linien eingezeichnet. Für den LogP wurde zusätzlich die entsprechende Untergrenze eingezeichnet. Die Bewertung erfolgt anhand der Ähnlichkeit zur Anfrage. Unter Hinzunahme weiterer Kriterien verringert sich die Ähnlichkeit der Produkte zum Anfragemolekül, insbesondere durch die Verwendung des Diversitätskriteriums. Lediglich die Produkte der abschließenden Bibliothek entsprechen allen Leitstrukturkriterien.

Bei theoretischen Beispielen ist es schwierig, Aussagen über die Relevanz der generierten Moleküle zu treffen, ohne diese synthetisieren und testen zu können. Hier besteht jedoch die Möglichkeit, auf bereits publizierte Struktur-Aktivitäts-Beziehungen (SAR) zurückzugreifen. Ashlanian und Mitarbeiter [192] geben für das Anfragemolekül einen K_i-Wert[1] von 4 nM an. Bei Schering Plough wurden hauptsächlich die Aktivität para-substituierter [(1H-Imidazol-4-yl)methyl] Benzamidine und Benzylamidine nachgewiesen [192]. Dies führte zur Entwicklung des 4-Benzyl-(1H-Imidazol-4-yl) Templates für H3-Rezeptor-Antagonisten [210]. Eine solche Substruktur ist im Fragmentraum zwar nicht enthalten, hat jedoch zum Beispiel zusammen mit einem 3,4-Dichlorophenyl (Reagenz 7 aus Abbildung 6.8 am zweiten Linker des Grundgerüstes) einen K_i von 7 nM.

Um die richtigen Substitutionsmuster zu erlangen, wird in der nächsten Iteration der Gewichtungsfaktor für die Regioselektivität auf 1,0 gesetzt. Die Phenylringe der Reagenzienauswahl für das R2-Linkatom des Grundgerüstes sind ebenso wie der korrespondierende Phenylring des Anfragemoleküls Sch79687 para-substituiert. Die ersten drei für das R3-Linkatom ausgewählten Reagenzien besitzen das korrekte 2,4-Substitutionsmuster, der Phenylring des vierten und fünften Reagenzes ist 3,4-, beziehungsweise 1,4-substituiert. Alle Reagenzien besitzen zumindest ein Chlor- beziehungsweise Fluoratom in meta-Stellung (Abbildung 6.9).

Um folglich eine diverse Reagenzienauswahl zu erhalten, wird als zusätzliche Restriktion lediglich ein Reagenz jedes Clusters zugelassen. Anderenfalls wird von der Bewertung der betroffenen Reagenzien 0,2 abgezogen. Bei der optimierten 5x5 Bibliothek (Abbildung 6.10) sind die Phenylringe der Reagenzienauswahl für das erste Linkatom weiterhin para-substituiert. Aufgrund der Restriktion werden für das erste Linkatom allerdings nur zwei 3,4-substituierte Phenylringe selektiert. Alle Reagenzien für das zweite Linkatom sind 2,4-substituiert. Zudem besitzen Reagenz 9 und 10 ein zusätzliches Fluoratom beziehungsweise eine Hydroxygruppe in para-Stellung. Die resultierenden Produkte sind chemisch unterschiedlicher und dennoch möglichst ähnlich zur Anfrage unter Berücksichtigung der Substitutionsmuster.

Um eine Bibliothek zu erhalten, bei der sich alle Produkte als Leitstruktur eignen, wird zusätzlich der Oprea-Filter verwendet. Die ausgewählten Reagenzien der optimierten Bibliothek aus Abbildung 6.11 sind teilweise identisch mit denen aus Abbildung

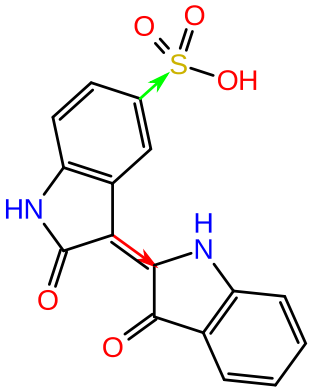
[1] Der K_i-Wert ist die Inhibitionskonstante, die die Stärke der Wechselwirkung zwischen Protein und Ligand beschreibt. Je kleiner der K_i-Wert, desto stärker bindet der Ligand [7].

6.10. Die Eigenschaftswerte aller Produkte entsprechen den gewünschten Bereichen, wie Abbildung 6.12 zeigt. Dabei ist zu beachten, dass die Bibliothek auf einer umgekehrten Zuordnung der Linkatome des Grundgerüstes beruht. Dies ist möglich, da beide Linkatome des Grundgerüstes kompatibel zu allen Reagenzien-Linktypen ist.

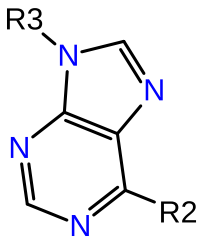
6.2.3 Cyclin-abhängige Kinase 2 (CDK2)

Die Cyclin-abhängige Kinase 2 (CDK2) ist ein bekanntes Zielprotein der Krebstherapie. Das Enzym spielt eine bedeutende Rolle in zwei Phasen des Zellzyklus [211, 212]. Aus der Literatur wurden zwei aktive Moleküle gewählt, um eine Purin-Bibliothek unter Verwendung des passenden Grundgerüstes zu optimieren. Die Bibliothek besteht aus einem Grundgerüst [50], mit zwei (siehe Abbildung 6.13b) beziehungsweise drei Linkatomen (siehe Abbildung 6.20b) sowie 11 653 beziehungsweise 29 649 Reagenzien.

6.2.3.1 Anfragemolekül Indirubin-5-Sulphonat (Beispiel CDK2-1)



(a) CDK2 Anfrage Indirubin-5-Sulphonat



(b) CDK2 Grundgerüst

Abbildung 6.13: Anfrage (a) und Grundgerüst (b) der CDK2-1-Bibliothek. (a) zeigt zudem die verwendeten Matching-Einschränkungen. Die Reagenzien dürfen ausschließlich auf die rot beziehungsweise grün markierten Bindungen gelegt werden.

Indirubin-5-Sulphonat [213] ist ein bekannter CDK2-Inhibitor. Als Grundgerüst wurde ein Purin mit zwei Linkatomen (R2 und R3) verwendet. 10 015 Reagenzien sind kompatibel zu dem Linkatom R2 und 1 638 Reagenzien zu dem Linkatom R3 des Grundgerüstes.

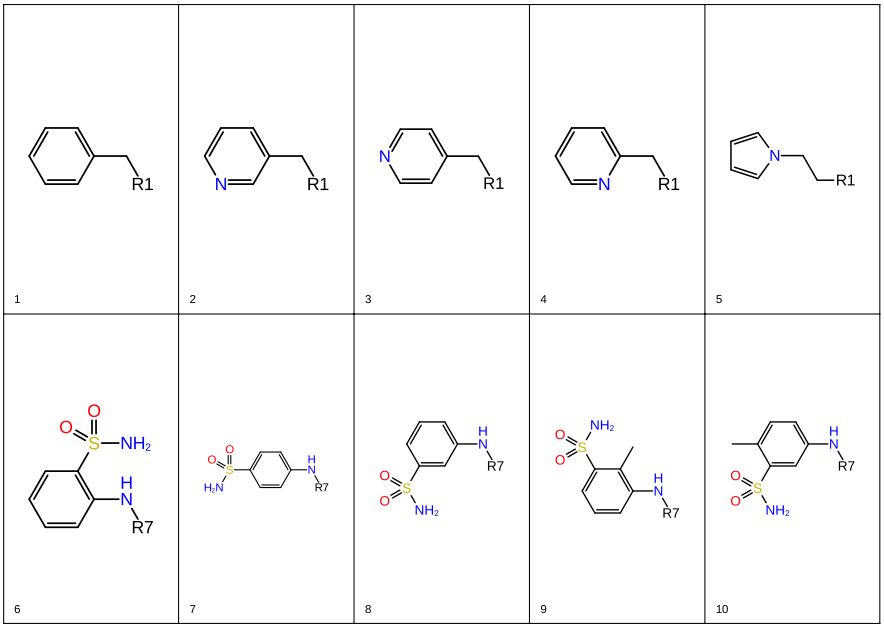


Abbildung 6.14: Subbibliothek CDK2-1-1 - 5x5 Bibliothek optimiert auf die Ähnlichkeit zur Anfrage.

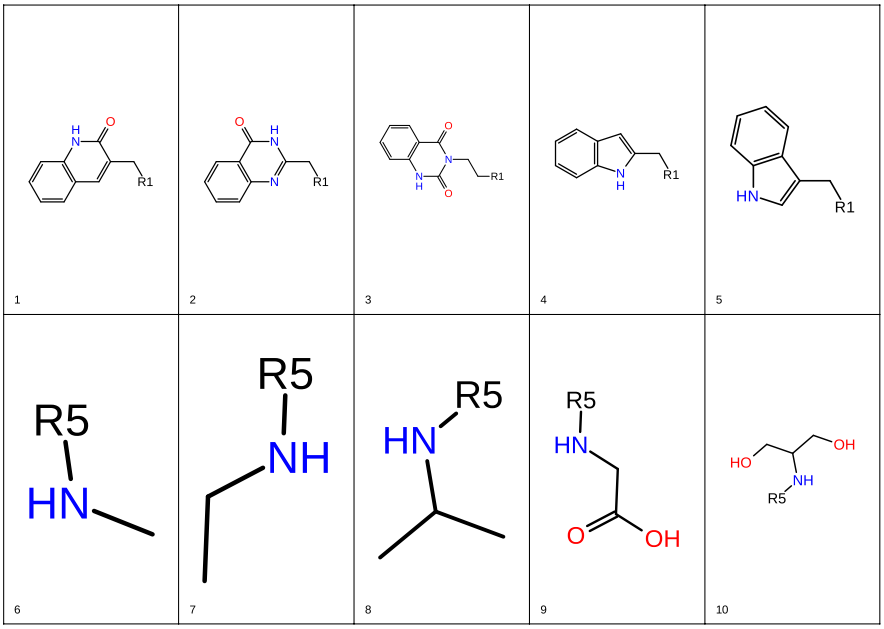


Abbildung 6.15: Subbibliothek CDK2-1-2 - 5x5 Bibliothek optimiert auf Ähnlichkeit zur Anfrage unter Verwendung eines anderen Initialwertes für den Zufallszahlengenerator. Die Auswahl der Reagenzien basiert auf einem alternativen Matching.

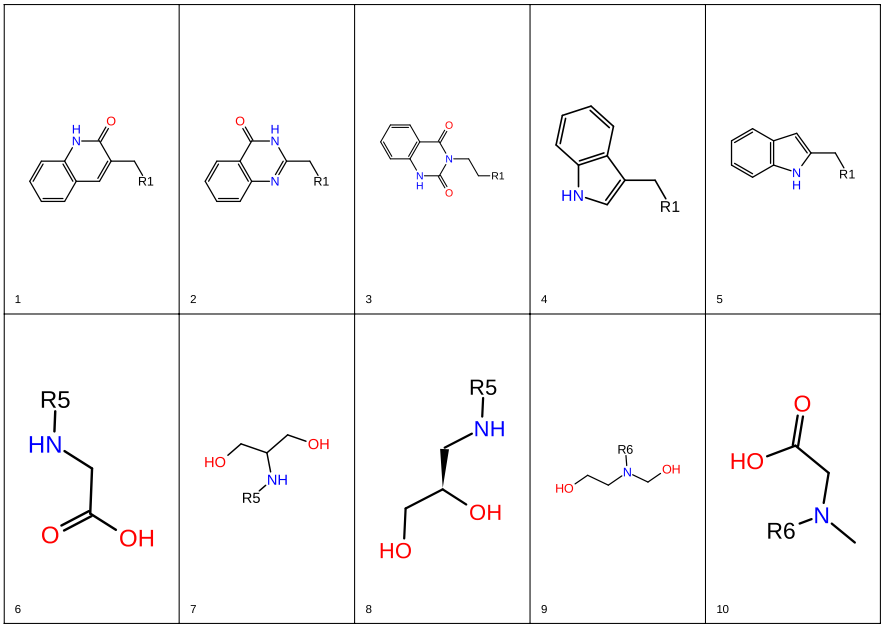


Abbildung 6.16: Subbibliothek CDK2-1-3 - Um ein konsistentes Matching zu erzielen, dürfen die zu R2 kompatiblen Reagenzien nur auf der grün eingefärbten Bindung platziert werden. Die Reagenzienauswahl für R3 bleibt dadurch unverändert.

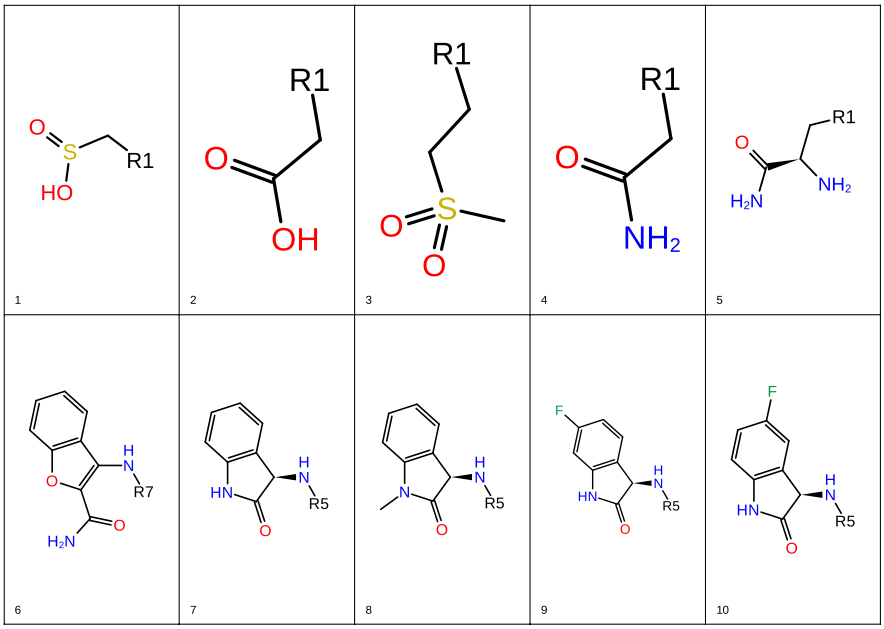
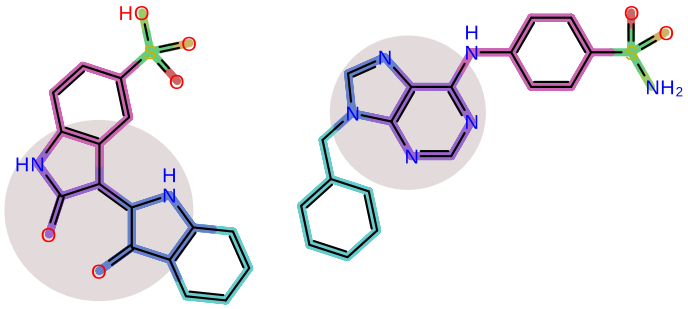
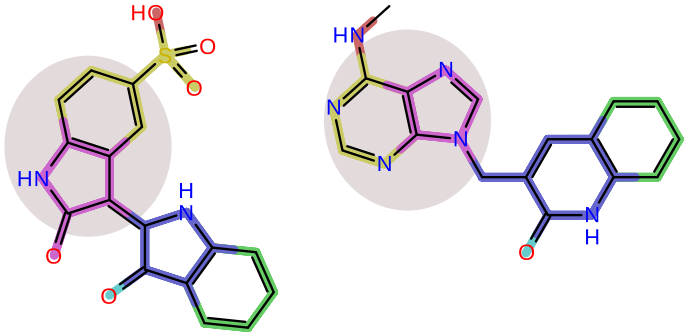


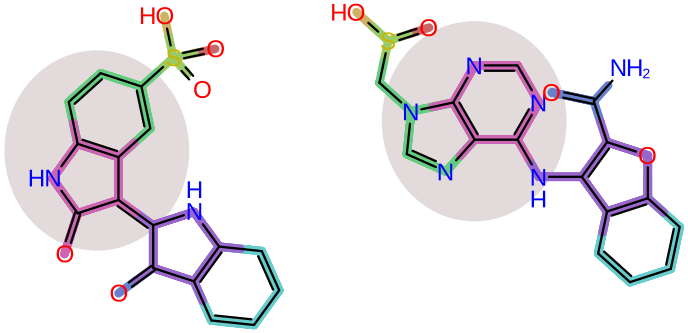
Abbildung 6.17: Subbibliothek CDK2-1-4 - Die Reagenzien, die kompatibel zu R2 sind, dürfen nur auf der rot eingefärbten Bindung platziert werden.



(a)



(b)



(c)

Abbildung 6.18: Alternative FTrees-Zuordnung des Anfragemoleküls mit dem jeweils besten Produkt der fokussierten Bibliotheken aus den Abbildungen 6.14, 6.15 und 6.17. Einander zugeordnete Substrukturen sind mit der gleichen Farbe hinterlegt. Die Zuordnung des Grundgerüstes ist zusätzlich mit einem grauen Kreis hervorgehoben. Die Produkte wurden jeweils aus dem Grundgerüst sowie den Reagenzien 1 und 6 generiert.

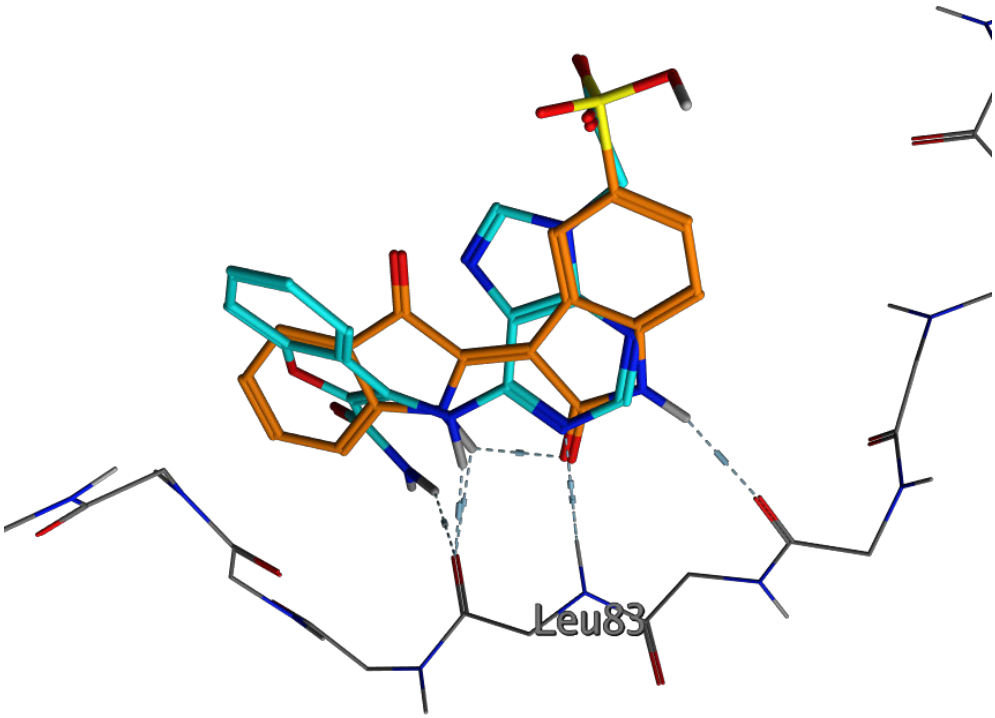


Abbildung 6.19: 3D-Überlagerung von Anfrage und Produkt in der Bindetasche von CDK2 - Zunächst wurden das Anfragemolekül (orange) und ein Produkt (blau) der Bibliothek CDK2-4 (Abbildung 6.17) mit ROCS [197] überlagert. Beide wurden mit MOE [214] in der Bindetasche von CDK2 (PDB-Code 1e9h) minimiert und binden an die Hinge-Region. Das Produkt wurde aus dem Grundgerüst sowie Reagenz 2 und 6 generiert. Im weiteren Designprozess könnte das Grundgerüst so modifiziert werden, dass die zusätzliche Wasserstoffbrücke, die das Anfragemolekül ausbildet, ebenfalls von den Produkten ausgebildet wird.

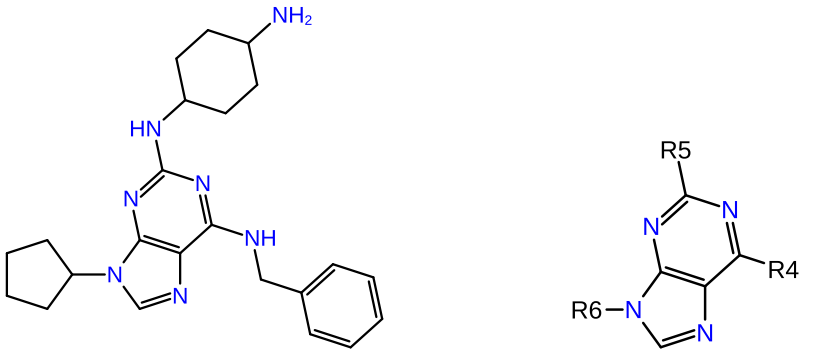
Wird eine fokussierte Bibliothek generiert, deren Produkte ähnlich zu dem Anfragemolekül sind, kann das Grundgerüst theoretisch beiden Indolinon-Substrukturen zugeordnet werden. Ein weiteres in Betracht kommendes Matching basiert auf der Platzierung des Grundgerüstes auf den beiden Fünfringen der Anfrage. Dies ist möglich, da die Ringe jeweils einem eigenen Knoten zugeordnet werden und an den verbindenden Kanten keine zusätzliche Information über die Art der Konnektivität gespeichert wird. Des Weiteren kann das Grundgerüst so auf die Substrukturen gelegt werden, dass der Fünfring des Grundgerüstes auf einem der beiden Sechsringe der Anfrage liegt. Dies führt zu weiteren alternativen Zuordnungen (siehe Abbildung 6.18).

Bei der Generierung einer fokussierten Bibliothek versucht der Optimierungsalgorithmus die Reagenzien für eine spezifische Zuordnung des Grundgerüstes auszuwählen. Dadurch wird die Ähnlichkeit aller entstehenden Produkte zur Anfrage verbessert. So resultiert die Optimierung mit den Standardparametern in die Bibliothek aus Abbildung 6.14, bei der das Grundgerüst den beiden Fünfringen zugeordnet wird (siehe Abbildung 6.18a). Dies ist, wie oben erwähnt, aus Sicht des Deskriptors nachvollziehbar, vor allem da die physikochemischen Eigenschaften der Knoten besser zueinander passen. Wird dagegen ein anderer Initialwert für den Pseudo-Zufallszahlengenerator verwendet, resultiert dies in einer Bibliothek (Abbildung 6.15), die auf einer alternativen Zuordnung des Grundgerüstes beruht, wie Abbildung 6.18b exemplarisch zeigt. Die Sulfonsäure der Anfrage ist in diesem Fall nur partiell den Reagenzien 6, 7 und 8 zugeordnet. Erfolgt eine Einschränkung des Matchings, so dass die Reagenzien die mit R2 verbunden werden können, nur der Kante zugeordnet werden dürfen, die durch den grünen Pfeil in Abbildung 6.13a repräsentiert wird, kann eine bessere Reagenzienauswahl gefunden werden (Abbildung 6.16). Die Auswahl der Reagenzien für R3 ändert sich dadurch nicht.

Im Gegensatz zum Anfragemolekül fehlt den erzeugten Produkten der Wasserstoffbrückendonor, um eine zweite Wasserstoffbrücke zur Hinge-Region auszubilden. Um Produkte zu generieren, die diese Eigenschaft erfüllen, werden die möglichen Zuordnungen eingeschränkt. Die Linkkanten der Reagenzien, die kompatibel zu R2 sind, dürfen nur auf die FTree-Kante gelegt werden, die durch den roten Pfeil in Abbildung 6.13a repräsentiert wird. Die resultierende Bibliothek zeigt Abbildung 6.17. Exemplarisch verdeutlicht Abbildung 6.19, dass die Produkte dieser Bibliothek den gewünschten Pharmakophor besitzen. Alternativ kann die Zuordnung der zu R3 kompatiblen Reagenzien auf

die grünen Kanten beschränkt werden. In beiden Fällen werden Bibliotheken generiert, die auf dieser Zuordnung basieren (siehe beispielhaft Abbildung 6.18c).

6.2.3.2 Anfragemolekül ZINC3591113 (Beispiel CDK2-2)



(a) CDK2 Anfrage ZINC3591113 **(b)** CDK2 Grundgerüst

Abbildung 6.20: Anfrage (a) und Grundgerüst (b) der zweiten CDK2-Bibliothek.

Ein weiterer CDK2-Inhibitor ist ZINC3591113 (Abbildung 6.20a). Er ist Teil des DUD-Datensatzes [202]. Das zur Optimierung verwendete Grundgerüst (Abbildung 6.20b) hat drei Linkatome. Dabei sind 6 159 Reagenzien kompatibel zu R4 und 24 477 Reagenzien kompatibel zu R5 und R6. Im Folgenden werden für jedes Linkatom vier Reagenzien gewählt, was zu 64 Produkten führt. Die Reagenzien 1-4 wurden für R6, die Reagenzien 5-8 für R5 und die Reagenzien 9-12 für R4 ausgewählt.

Abbildung 6.21 zeigt eine Subbibliothek, deren alleiniges Optimierungskriterium die Ähnlichkeit zum Anfragemolekül ZINC3591113 ist. Die physikochemischen Eigenschaften der entstehenden Produkte entsprechen nicht den Kriterien für Leitstrukturen (siehe Abbildung 6.25), was sich daraus ergibt, dass auch das Anfragemolekül diesen nicht genügt. Zu Verbesserung des Profils wird im nächsten Schritt ein Produktfilter mit den Oprea-Kriterien verwendet.

Die resultierende Bibliothek CDK2-2-2 (Abbildung 6.22) ist suboptimal. Für R4 und R5 wurden sowohl aromatische als auch nicht-aromatische Ringe gewählt. Dies liegt daran, dass die Substitutionsstellen R4 und R5 für den FTree-Deskriptor nicht unterscheidbar sind und durch den Filter viele Produkte abgelehnt werden. Dadurch wird erst spät eine erlaubte Lösung gefunden. Weitere Optimierungsschritte sind notwendig,

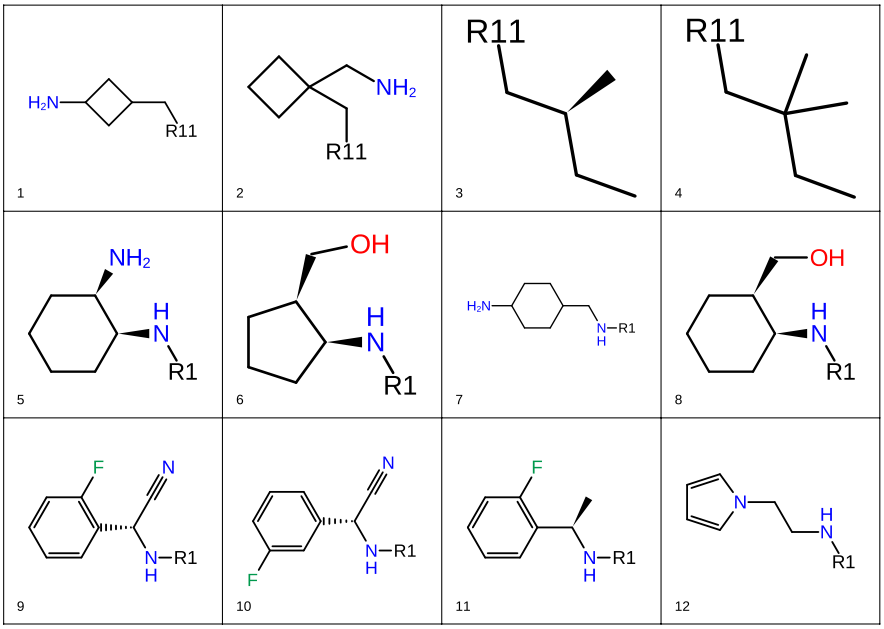


Abbildung 6.21: Subbibliothek CDK2-2-1 - 4x4x4 Bibliothek optimiert auf die Ähnlichkeit zu ZINC3591113.

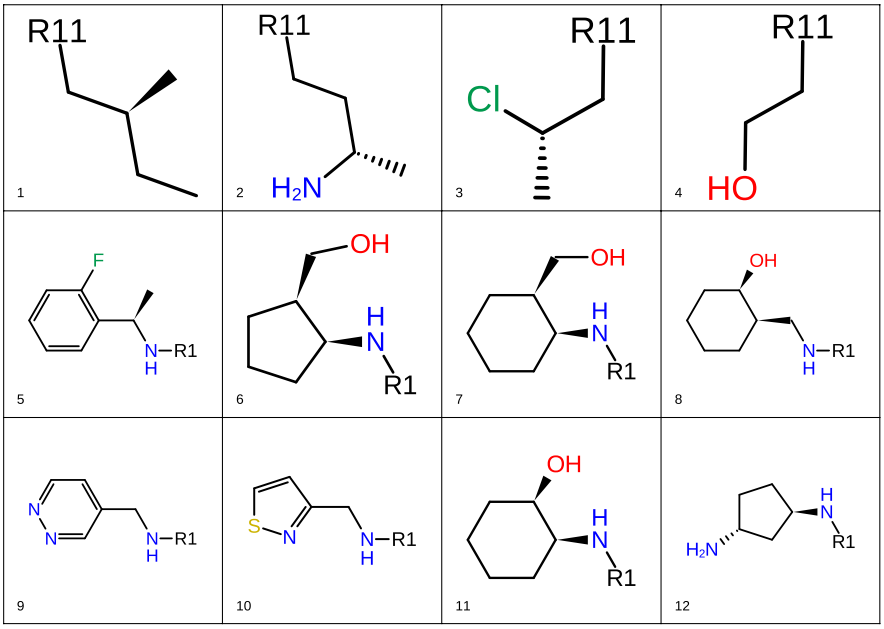


Abbildung 6.22: Subbibliothek CDK2-2-2 - 4x4x4 Bibliothek, bei deren Generierung zusätzlich der Oprea-Filter verwendet wurde.

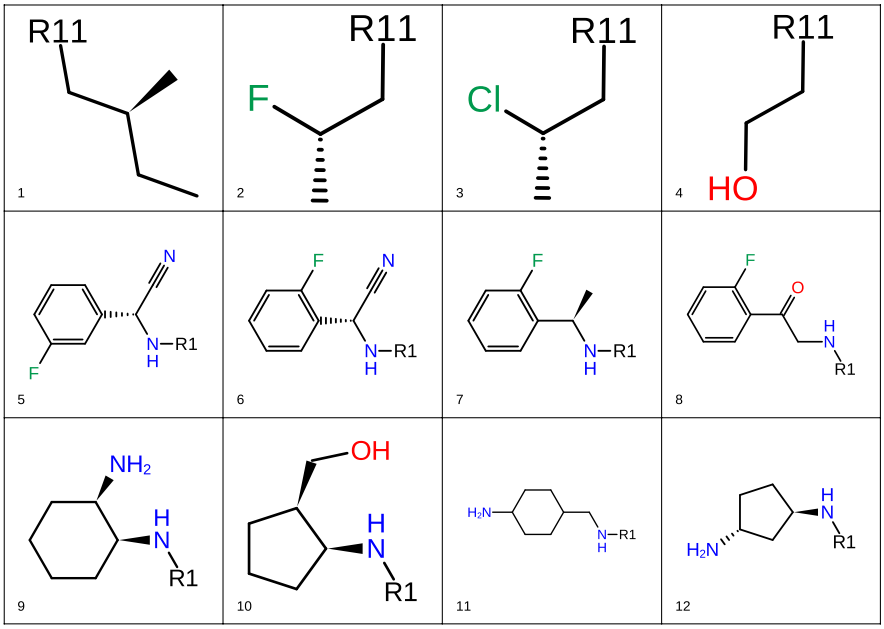


Abbildung 6.23: Subbibliothek CDK2-2-3 - 4x4x4 Bibliothek, bei deren Generierung die Oprea-Kriterien in die Bewertungsfunktion integriert wurden, statt den Filter zu verwenden.

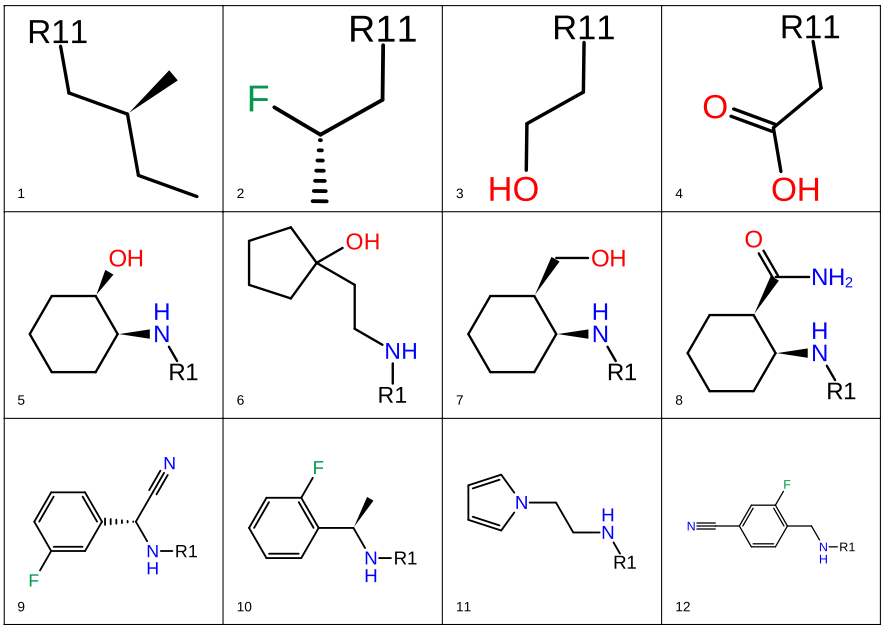


Abbildung 6.24: Subbibliothek CDK2-2-4 - 4x4x4 Bibliothek, bei der die Oprea-Kriterien in die Bewertungsfunktion integriert wurden und zusätzlich nur ein Reagenz pro Cluster erlaubt ist.

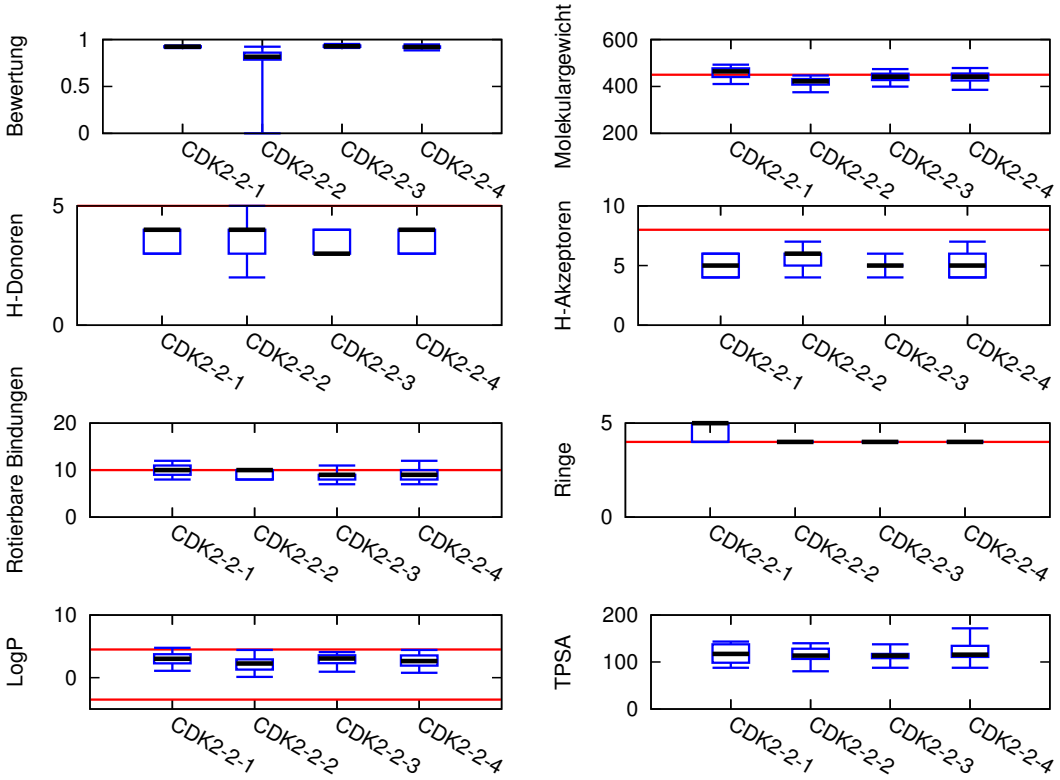


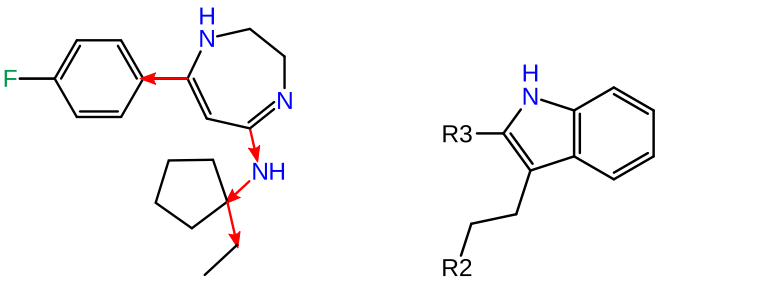
Abbildung 6.25: Eigenschaftsprofile der generierten Bibliotheken dargestellt durch Box-Whisker Plots. Die Obergrenzen des Oprea-Filters sind als rote Linien eingezeichnet. Eine zusätzliche Untergrenze existiert für den LogP. Nur bei Verwendung des Filters entsprechen alle Produkte den Leitstrukturkriterien.

damit der Algorithmus konvergiert. Dennoch besitzen alle Produkte die gewünschten Eigenschaften.

Statt den Produktfilter zu verwenden, können die Oprea-Kriterien in die Bewertungsfunktion aufgenommen werden. Exemplarisch werden die Eigenschaften mit 0,05 und die Ähnlichkeit zur Anfrage mit 0,7 gewichtet. Die meisten der 64 Produkte der Bibliothek CDK2-2-4 zeigen die gewünschten Eigenschaften, einige wenige besitzen hingegen ein zu hohes Molekulargewicht oder zu viele rotierbare Bindungen (siehe auch Abbildung 6.25).

Um die Reagenzienauswahl schließlich stärker zu diversifizieren, wird lediglich ein Reagenz aus jedem Ähnlichkeitscluster erlaubt. Anderenfalls werden die betroffenen Reagenzien durch einen Abzug von 0,2 von der Bewertung bestraft. Dies resultiert in die Bibliothek CDK2-2-4 (Abbildung 6.24). Auch hier gibt es einige wenige Produkte, die nicht allen gewünschten Kriterien entsprechen. Auffällig ist jedoch, dass die aromatischen Ringe für R4 ausgewählt wurden. Im vorliegenden Fall können die Reagenzienlisten ausgetauscht werden. Dennoch zeigt auch dieses Beispiel, wie wichtig die Einschränkung der Matches und die Regioselektvität sein können, um in einem solchen Fall die Ergebnisse zu beeinflussen und zu verbessern.

6.2.4 Serotonin-5-HT$_{2A}$-Rezeptor



(a) 5-HT$_{2A}$ Anfrage **(b)** 5-HT$_{2A}$ Grundgerüst

Abbildung 6.26: Anfrage (a) und Grundgerüst (b) der 5-HT$_{2A}$-Rezeptor Bibliothek. (a) zeigt zudem die Matching-Restriktionen für das Anfragemolekül, die eingeführt wurden, um eine bestimmte Ausrichtung der Knotenzuordnung zu garantieren. Ausschließlich die gerichteten FTree-Kanten, die als rote Pfeile auf den korrespondierenden Bindungen eingezeichnet sind, können einer Reagenzien-Linkkante zugeordnet werden.

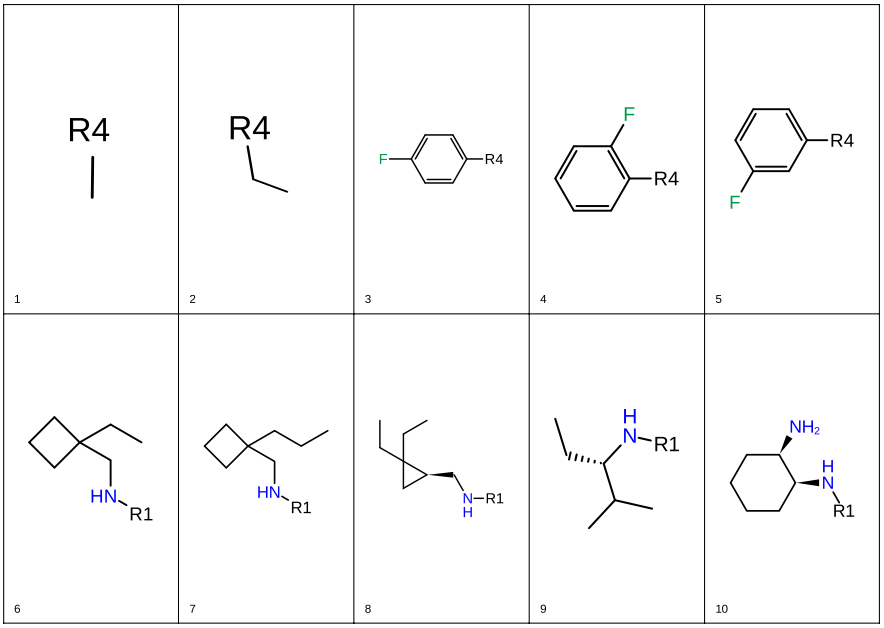


Abbildung 6.27: Subbibliothek 5-HT$_{2A}$-1 -Die 5x5 Bibliothek wurde optimiert auf die Ähnlichkeit zum Anfragemolekül.

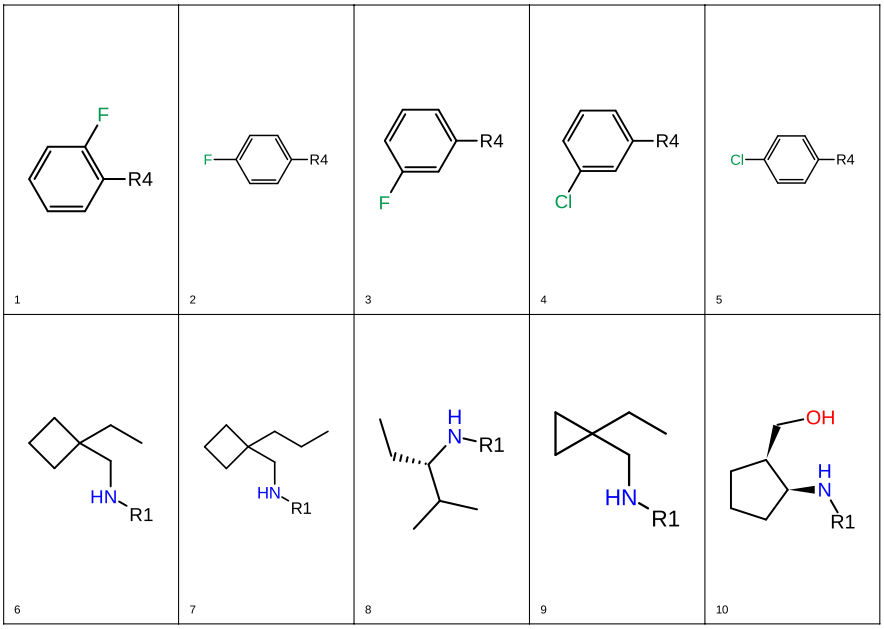


Abbildung 6.28: Subbibliothek 5-HT$_{2A}$-2 - 5x5 Bibliothek optimiert auf die Ähnlichkeit zum Anfragemolekül. Die möglichen Zuordnungen der Reagenzien-Links wurden eingeschränkt (Abbildung 6.26a).

Der 5-HT_{2A} Rezeptor gehört zur Familie der Serotonin-Rezeptoren (5-Hydroxytryptamin). Er ist ein *G-Protein-gekoppelter Rezeptor (GPCR)* und kontrolliert die Ausschüttung von Neurotransmittern. Aus diesem Grund ist er für die pharmazeutische Forschung von Interesse. So werden 5-HT_{2A}-Rezeptor-Antagonisten zum Beispiel entwickelt um Depression, Schizophrenie und Schlaflosigkeit zu therapieren [215]. Ein Indol-Grundgerüst [216] (siehe Abbildung 6.26b) sowie 25 567 Reagenzien werden zum Entwurf der in diesem Fallbeispiel aufgeführten fokussierten Bibliotheken genutzt. Dabei sind 23 486 Reagenzien kompatibel zu Linkatom R2 sowie 2 081 Reagenzien zu Linkatom R3 des Grundgerüstes. Abbildung 6.26a zeigt das verwendete Anfragemolekül [215].

Zunächst wird eine 5x5 Bibliothek generiert, deren Produkte möglichst ähnlich zum Anfragemolekül sein sollen (Abbildung 6.27). Die resultierende Bibliothek basiert auf zwei alternativen Matchings, wie Abbildung 6.29 verdeutlicht. Bei Matching *a* wird der Sechsring des Indols dem Sechsring der Anfrage zugeordnet. Der Algorithmus selektiert möglichst kleine Reagenzien, da diese nicht zugeordnet werden. Beim zweiten Matching wird der Sechsring eines Reagenzes auf den Sechsring der Anfrage gelegt. Beide Zuordnungen ergeben hohe FTree-Ähnlichkeitswerte. Auch die dargestellten 3D-Überlagerungen zeigen jeweils eine geringe räumliche Abweichung.

Oftmals hat der Anwender jedoch eine bestimmte Überlagerung im Sinn. Dies ist insbesondere der Fall, wenn das Grundgerüst anhand der 3D-Überlagerung von Anfragemolekül und einem aktiven Molekül dieser Strukturklasse ausgewählt wurde. Die Auswahl der Reagenzien sollte dann ebenfalls auf dieser Überlagerung basieren. Für die folgende Optimierung werden die möglichen Zuordnungen deshalb so eingeschränkt, dass die Ausrichtung des Grundgerüstes wie gewünscht festgelegt wird. Die Linkkanten der Reagenzien können lediglich den Kanten, die in Abbildung 6.26a rot eingefärbt sind, zugeordnet werden. Dies resultiert in einer Bibliothek, die ausschließlich diese Zuordnung zeigt (Abbildung 6.28). Die fünf gewählten Reagenzien für das Linkatom R3 enthalten somit nur Fluor- und Chlor-Substituenten, jedoch wird das Substitutionsmuster dabei nicht beachtet.

Aus diesem Grund wird die Regioselektivität für die folgende Bibliothek mit einer Gewichtung von 1.0 angewendet. Die resultierenden Reagenzien weisen ein Fluor-, Chlor- oder Bromatom beziehungsweise eine Ethinyl-Gruppe in para-Stellung, beziehunsweise ein Fluoratom in meta-Stellung, auf. Die gewählten Reagenzien für das zweite

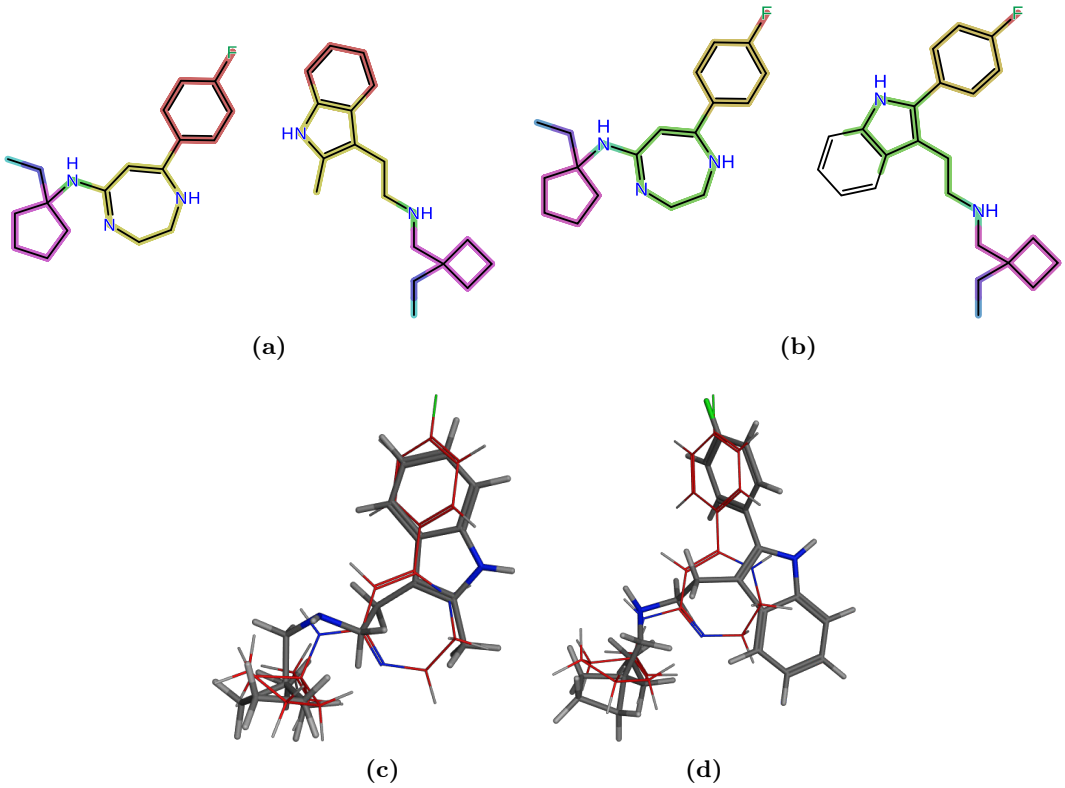


Abbildung 6.29: Feature-Tree-Matching (a und b) und 3D-Superpositionierung (c beziehungsweise d) von Anfragemolekül und den Produkten, die sich aus der Verknüpfung von Grundgerüst sowie den Reagenzien 1 und 6 beziehungsweise 4 und 6 ergeben. Die Superpositionierung mit FlexS [194] basiert auf den FTree-Matchings.

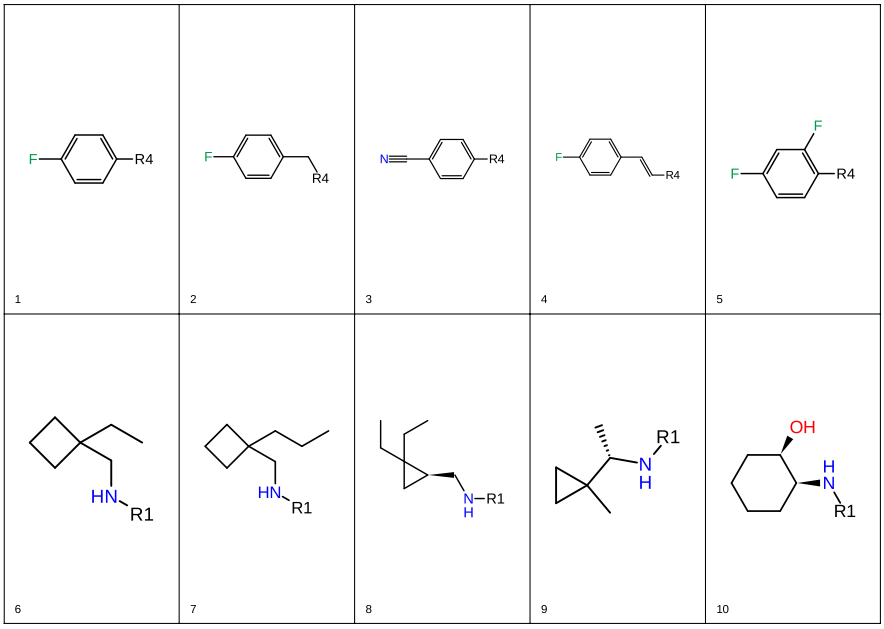


Abbildung 6.30: Subbibliothek 5-HT$_{2A}$-4 - 5x5 Bibliothek, bei der zusätzlich lediglich ein Reagenz pro Cluster erlaubt ist. Durch den Gewichtungsfaktor für die Regioselektivität sind die Reagenzien für Linkatom R3 weiterhin para-substituiert.

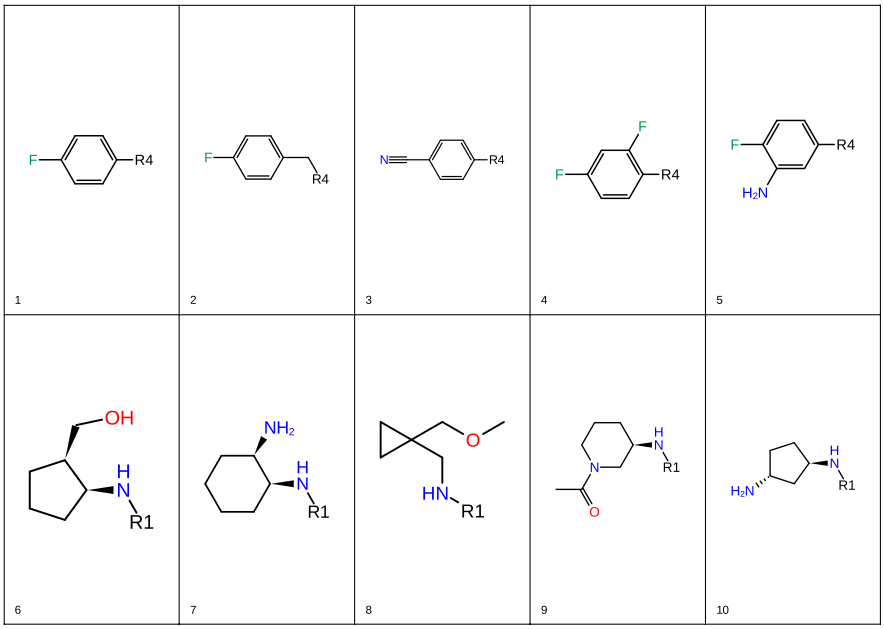


Abbildung 6.31: Subbibliothek 5-HT$_{2A}$-5 - 5x5 Bibliothek, bei der die resultierenden Produkte zusätzlich den Oprea-Kriterien entsprechen.

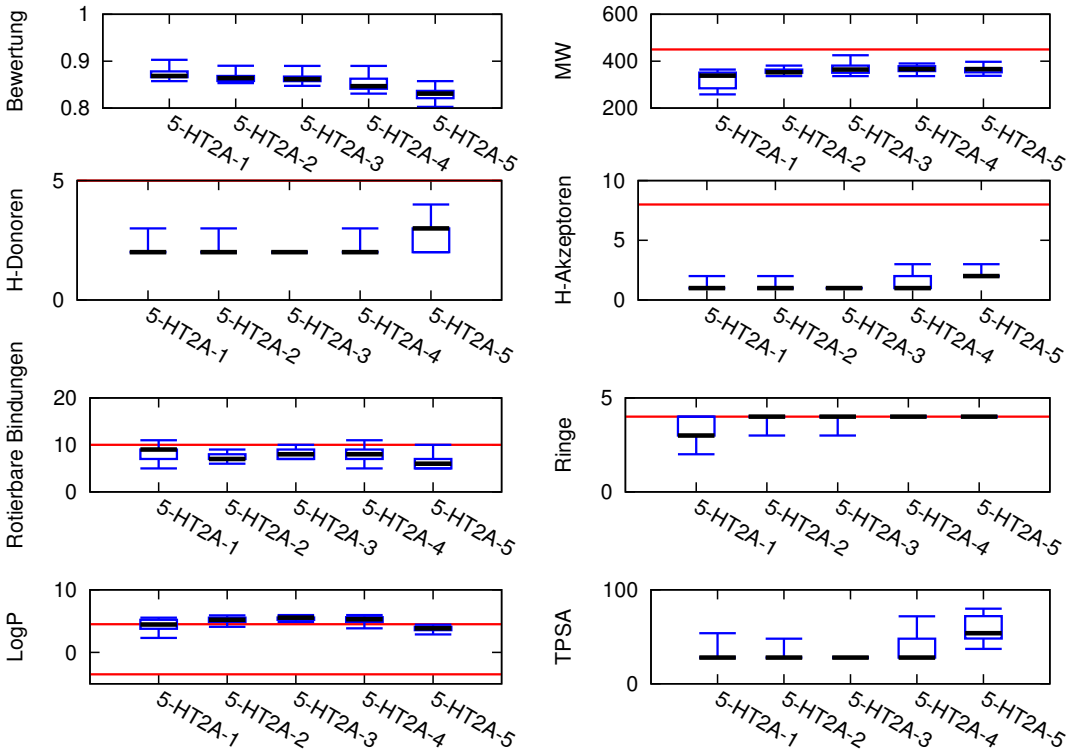


Abbildung 6.32: Eigenschaftsprofile der generierten Bibliotheken dargestellt durch Box-Whisker Plots. Die Obergrenzen des Oprea-Filters wurden als rote Linien eingezeichnet, für die LogP Werte gibt es eine zusätzliche Untergrenze. Unter Hinzunahme weiterer Kriterien verringert sich die Ähnlichkeit der Produkte zum Anfragemolekül. Durch die Hinzunahme des Oprea-Filters entstehen Produkte, die immer noch ähnlich zur Anfrage sind und zusätzlich den Leitstrukturkriterien entsprechen.

Linkatom des Grundgerüstes (R2) bestehen aus sekundären Aminen mit einem Ring, der auf den Fünfring der Anfrage gelegt werden kann. Wird eine Vorsortierung der Reagenzien anhand ihrer Ähnlichkeit zu einer Substruktur der Anfrage verwendet, unterscheidet sich die Lösung lediglich anhand eines Reagenzes. Auch die Anfangslösung, die innerhalb von Sekunden generiert wird, unterscheidet sich nur anhand eines Reagenzes. Im ersten Fall liegt es an der stochastischen Optimierung. Im zweiten Fall liegt es an der Tatsache, dass bei Betrachtung der Produkte die Ähnlichkeit über die Größe der Feature-Trees normalisiert wird. Dadurch wirkt sich ein Nullmatch negativer aus als die fehlende Ringeigenschaft.

Um schließlich Bibliotheken zu erhalten, die eine möglichst diverse Auswahl an Reagenzien beinhalten, wird zusätzlich nur ein Reagenz aus jedem Cluster erlaubt. Alle Reagenzien für R3 sind weiterhin para-substituiert, wie Abbildung 6.30 zeigt. Die Bibliothek entspricht allerdings nicht den Leitstruktur-Kriterien in Bezug auf die Anzahl der rotierbaren Bindungen und der berechneten LogP-Werte (siehe Abbildung 6.32).

Unter Verwendung des Oprea-Filters resultiert die Reagenzienauswahl in Produkte (siehe Abbildung 6.31), deren Eigenschaften in den gewünschten Bereichen liegen (Abbildung 6.32). Um den berechneten LogP zu verringern, enthalten die Reagenzien für Linkatom R2 zusätzlich ein Stickstoff- oder Sauerstoffatom. Aufgrund des Ähnlichkeits- und des Regioselektivitätkriteriums sind die Produkte dennoch ähnlich zur Anfrage und die Reagenzien für Linkatom R3 para-substituiert.

Die multikriterielle Optimierung erzielt die gewünschten Ergebnisse. Zwar verringert sich der berechnete Qualitätswert der Bibliothek durch die Hinzunahme weiterer Kriterien, da die durchschnittliche Ähnlichkeit zur Anfrage sinkt. Die Eigenschaften der resultierenden Produkte entsprechen jedoch den Leitstrukturkriterien des Oprea-Filters. Zudem besteht die Bibliothek aus möglichst diversen Reagenzien, wobei dennoch alle entstehenden Produkte ähnlich zur Anfrage sind. Nicht zuletzt ist das Feature-Tree-Matching konsistent und regiosensitiv.

6.2.5 Analyse von Laufzeit und Speicherbedarf

In diesem Abschnitt wird die Laufzeit und der Speicherbedarf anhand der einzelnen Fallbeispiele betrachtet. Es wird untersucht, wie sich die Anwendung der einzelnen Kriterien auswirken. Abschließend wird die Abhängigkeit der Laufzeit von Bibliotheksgröße und FTree-Generierungsmodus analysiert.

6.2.5.1 Fallbeispiel H3

Subbibliothek	Hinzugefügtes Optimierungskriterium	Laufzeit
H_3-1	Ähnlichkeit zur Anfrage	8:09 min
H_3-2	Regioselektivität	7:56 min
H_3-3	Maximal ein Reagenz pro Cluster	8:11 min
H_3-4	Leitstruktur-Filter	6:25 min

Tabelle 6.6: Laufzeiten bei der Optimierung der H_3-Bibliothek.

Bei dem H3-Fallbeispiel erhöht sich unter Verwendung des neuen Modus zur Feature-Tree-Generierung die Laufzeit von 1 Minute und 40 Sekunden auf 8 Minuten und 9 Sekunden. Im Gegensatz zum alten Modus der Feature-Tree-Generierung erhöht sich die Anzahl der Knoten des Grundgerüst-FTrees von drei auf vier. Unter Verwendung der Regioselektivität sowie des Filters bricht die Optimierung vorzeitig nach 191 486 beziehungsweise 185 639 Schritten ab, da während der letzten 50 000 Schritte keine Verbesserung mehr erfolgte. Die Laufzeit der Optimierung ist von Regioselektivitätberechnungen und dem Diversitätskriterium nahezu unbeeinflusst (Tabelle 6.6). Der Filter reduziert zwar die Laufzeit, bei dem verwendeten Fragmentraum ist die Auswahl der Reagenzien jedoch bereits auf die Problemstellung zugeschnitten. Wird stattdessen der Raum genutzt, der zur Validierung in der Publikation [32] gewählt wurde, ergibt sich durch den Filter eine Reduktion der Laufzeit um die Hälfte von 8 Minuten und 9 Sekunden auf 3 Minuten und 55 Sekunden. Da eine Vielzahl der betrachteten Produkte den Kriterien nicht genügen, werden entsprechend weniger Ähnlichkeitsvergleiche durchgeführt.

Der Speicherbedarf einer Optimierung liegt bei diesem Fallbeispiel bei ungefähr 400 MB. Neben den 330 MB für den Fragmentraum werden zusätzlich ungefähr 70 MB für die Deskriptoren, insbesondere die Feature-Trees und die Vergleichsmatrizen benötigt.

6.2.5.2 Fallbeispiel CDK2-1

Tabelle 6.7 zeigt die Laufzeiten bei der Generierung der unterschiedlichen Bibliotheken. Durch die Restriktionen werden jeweils 13 Sekunden (CDK2-1-2 versus CDK2-1-3) beziehungsweise 15 Sekunden (CDK2-1-1 versus CDK2-1-4) mehr benötigt. Allerdings unterscheidet sich die Laufzeit bei der Generierung von CDK2-1-1 und CDK2-1-2 ebenfalls um 11 Sekunden. Da der einzige Unterschied im Initialwert des Zufallszahlengenerators

Subbibliothek	Hinzugefügtes Optimierungskriterium	Laufzeit
CDK2-1-1	Ähnlichkeit zur Anfrage	6:08 min
CDK2-1-2	Veränderter Initialwert des Zufallszahlengenerators	5:57 min
CDK2-1-3	Matching-Restriktion	6:10 min
CDK2-1-4	Alternative Matching-Restriktion	6:23 min

Tabelle 6.7: Laufzeiten bei der Optimierung der CDK2-1-Bibliothek.

liegt, können die Laufzeitunterschiede in dieser Größenordnung auf die stochastische Traversierung des Suchraums und die dabei betrachteten Matchings zurückgeführt werden. Der Speicherbedarf liegt bei 340 MB für den Fragmentraum und bei 65 MB für die Deskriptoren und Vergleichsmatrizen.

6.2.5.3 Fallbeispiel CDK2-2

Subbibliothek	Hinzugefügtes Optimierungskriterium	Laufzeit
CDK2-2-1	Ähnlich zu Anfrage	19:14 min
CDK2-2-2	Leitstruktur-Filter	1:19 min
CDK2-2-3	Leitstrukturkriterien in der Bewertungsfunktion, kein Filter	21:38 min
CDK2-2-4	Maximal ein Reagenz pro Cluster	21:31 min

Tabelle 6.8: Laufzeiten bei der Optimierung der CDK2-2-Bibliothek.

Im Gegensatz zu der Generierung von 5x5 Bibliotheken ist die Laufzeit bei der Generierung von 4x4x4 Bibliotheken deutlich erhöht, da sie in 64 statt 25 Produkte resultieren. Umso deutlicher zeigt sich die Reduktion der Laufzeit durch die Anwendung eines Eigenschaftsfilters. Entspricht ein Produkt nicht den Kriterien, entfällt der Ähnlichkeitsvergleich. Werden die Eigenschaftskriterien stattdessen in die Bewertungsfunktion integriert, kann keine Laufzeitreduktion erzielt werden, da alle Ähnlichkeitsvergleiche ausgewertet werden müssen. Die Hinzunahme des Diversitätskriteriums führt zu keiner signifikanten Veränderung der Laufzeit. Neben den 560 MB für die Speicherung der Fragmente werden 260 MB für die Speicherung von Deskriptoren und Vergleichsmatrizen benötigt.

6.2.5.4 Fallbeispiel $5HT_{2A}$

Subbibliothek	Hinzugefügtes Optimierungskriterium	Laufzeit
5-HT_{2A}-1	Ähnlichkeit zur Anfrage	4:45 min
5-HT_{2A}-2	Eingeschränktes Matching	4:15 min
5-HT_{2A}-3	Regioselektiv und vorsortiert (Startlösung)	0:03 min
5-HT_{2A}-4	Maximal ein Reagenz pro Cluster	3:57 min
5-HT_{2A}-5	Zusätzliche Verwendung des Leitstruktur-Filters	1:52 min

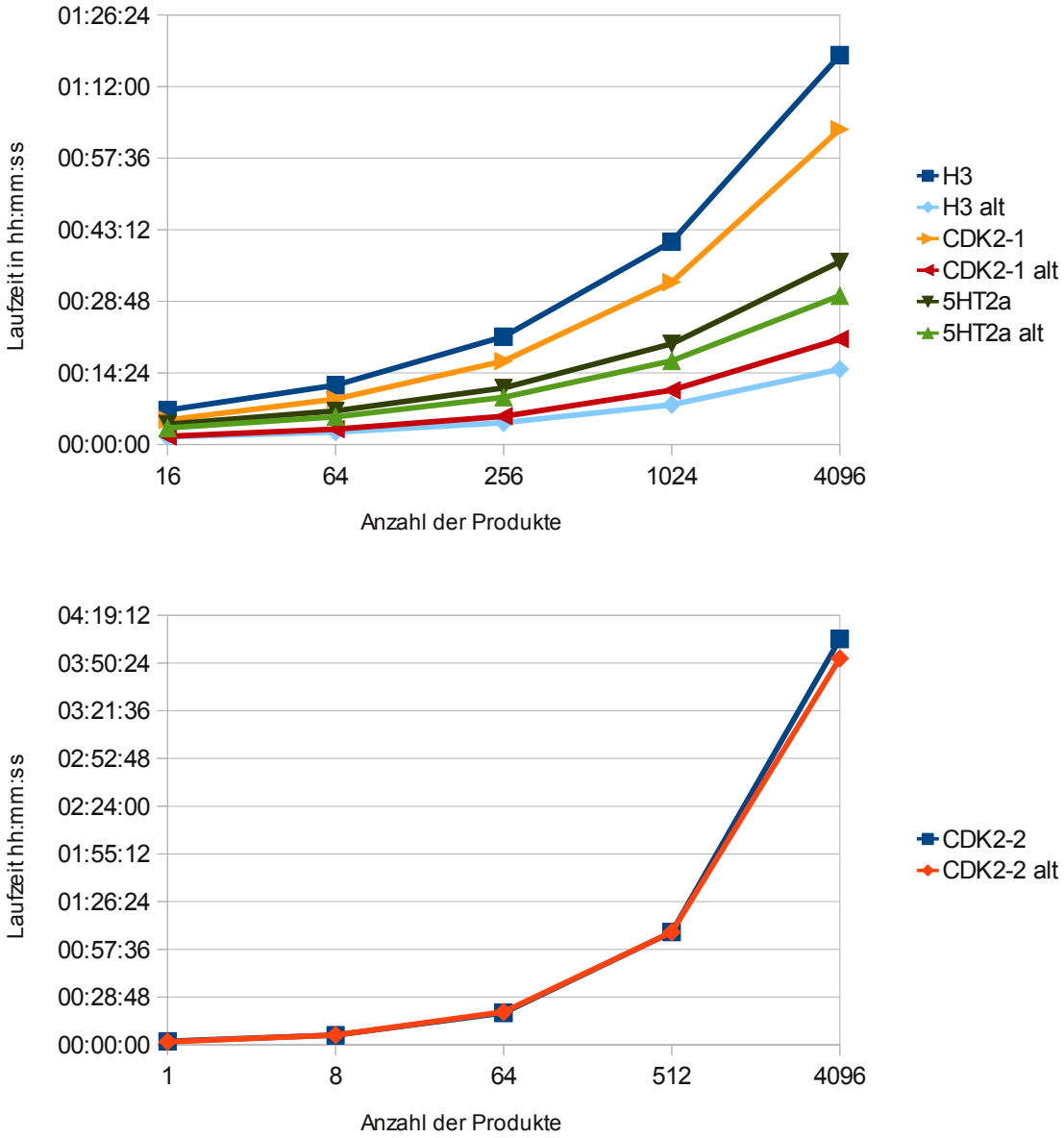
Tabelle 6.9: Laufzeiten bei der Optimierung der 5-HT_{2A}-Bibliothek unter Hinzunahme unterschiedlicher Kriterien.

Werden die Laufzeiten bei der Optimierung von $5HT_{2A}$ betrachtet (siehe Tabelle 6.9), zeigt sich, dass durch die Anwendung des Regioselektivitätskriteriums die Laufzeit positiv beeinflusst wird. Da an aromatischen Ringen keine Erweiterungsmatches mehr möglich sind, ist es erforderlich, dass sich der Match-Search-Algorithmus direkt rekursiv aufruft. Insgesamt werden dadurch weniger mögliche Matches betrachtet. Des Weiteren kann durch die Einschränkung der möglichen Matchings und die Vorsortierung der Reagenzien in wenigen Sekunden eine Bibliothek ermittelt werden, deren Produkte alle auf dem gewünschten Matching beruhen. Die Vorsortierung liefert somit eine gute Startauswahl der Reagenzien für die multikriterielle Optimierung. Auch die Hinzunahme des Diversitätskriteriums wirkt sich nicht negativ auf die Laufzeit aus. Die verringerte Laufzeit lässt sich dagegen auf das Abbruchkriterium zurückführen, da die Optimierung nach 191 486 Schritten stoppt. Durch die verwendeten Filterkriterien reduziert sich die Laufzeit jedoch wesentlich, da keine Ähnlichkeitsvergleiche durchgeführt werden, wenn das resultierende Produkt nicht den Kriterien entspricht.

Für eine Optimierung werden ungefähr 830 MB Hauptspeicher benötigt. 630 MB werden für die Speicherung des Fragmentraumes und 200 MB für die Speicherung von Deskriptoren und Vergleichsmatrizen verwendet.

6.2.5.5 Abhängigkeit der Laufzeit von Bibliotheksgröße und FTree-Generierungsmodus

In diesem Abschnitt wird untersucht, welchen Einfluss die Größe der zu generierenden Subbibliothek auf die Laufzeit hat. Zusätzlich werden der neue und alte Feature-Tree-



(b) Grundgerüst mit drei Linkatomen

Abbildung 6.33: Abhängigkeit der Laufzeit von der Subbibliotheksgröße - Exemplarisch wurden für die Fallbeispiele Subbibliotheken unterschiedlicher Größe generiert. Die Laufzeiten bei unterschiedlichen Feature-Tree-Generierungsmodi werden für Bibliotheken mit einem Grundgerüst mit zwei (a) beziehungsweise drei (b) Linkatomen gegenübergestellt. Die terminalen Schweratome werden entweder einem eigenen Knoten (CDK2-2) oder dem Knoten des Bindungspartners zugeordnet (CDK2-2 alt). An der x-Achse ist die Anzahl der aus der Bibliothek resultierenden Produkte aufgeführt.

Generierungsmodus miteinander verglichen. Dafür wird eine Simulierte Abkühlung mit 200 000 Schritten durchgeführt. Alleiniges Optimierungskriterium ist jeweils die Ähnlichkeit zum Anfragemolekül. Die Abhängigkeit der Laufzeit von der Anzahl der Schritte ist linear. Zudem werden alle Vergleiche von Reagenz und Anfragesubstruktur nur einmal durchgeführt. Die Anzahl der verfügbaren Reagenzien wirkt sich somit nur in geringem Maße auf die Laufzeit aus. Die Größe des Fragmentraumes bestimmt jedoch den Speicherbedarf und die Laufzeit während der Präprozessierung (siehe auch Kapitel 6.1).

Abbildung 6.33a zeigt die Laufzeiten bei variabler Bibliotheksgröße für das H_3- (Kapitel 6.2.2), $5HT_{2A}$- (Kapitel 6.2.4) und eines der CDK2-Fallbeispiele (Kapitel 6.2.3.1). Die verwendeten Grundgerüste besitzen zwei Linkatome. Um 64 Produkte zu generieren, werden für jede Substitutionsstelle jeweils acht Reagenzien selektiert. Durch den Austausch eines Reagenzes während der Optimierung müssen acht Produkte neu bewerten werden. Abbildung 6.33b zeigt dagegen die Laufzeiten für das zweite der beiden CDK2-Fallbeispiele (Kapitel 6.2.3.2). Das dabei verwendete Grundgerüst hat drei Linkatome. Um 64 Produkte zu generieren, werden jeweils vier Reagenzien ausgewählt. Der Austausch eines Reagenzes führt somit zu einer Neubewertung von sechzehn Produkten. Für die gleiche Anzahl an resultierenden Produkten müssen in jeder Iteration mehr Bewertungen durchgeführt werden. Dies zeigt, dass die Laufzeit linear abhängig von der Anzahl der betrachteten Produkte während einer Iteration ist. Interessant ist auch der Einfluss der Knotenzuordnung bei der Generierung der Feature-Trees. Der Vergleich der FTree-Generierungsmodi zeigt deutlich eine Abhängigkeit der Laufzeit von der Anzahl der Feature-Tree-Knoten von Anfrage und Grundgerüst. Im Fall von $5HT_{2A}$ besitzt die Anfrage einen zusätzlichen Knoten, bei CDK2-1 hat der Anfrage-FTree vier zusätzliche Knoten, Bei H_3 kommen drei Knoten bei der Anfrage hinzu. Das Grundgerüst besitzt einen zusätzlichen Knoten. Hier vervierfacht sich die Laufzeit. Bei CDK2-2 kommen keine weiteren Knoten hinzu, somit ist die Laufzeit für beide Modi ähnlich. Allerdings ist sie nicht identisch, da die Reagenzien-FTrees unterschiedlich generiert wurden.

Für alle Fallbeispiele führen die unterschiedlichen FTree-Generierungsmodi zu gleichwertigen Bibliotheken. Werden zudem die Ergebnisse der Anreicherungsstudien betrachtet (Kapitel 6.1.2), ist es empfehlenswert, den neuen FTree-Generierungsmodus nur zu verwenden, wenn eine regioselektive Optimierung durchgeführt wird oder nach alterna-

tiven Zuordnungen gesucht wird. Werden kleine Bibliotheken generiert, erlauben jedoch beide Modi ein nahezu interaktives Vorgehen.

7 Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurde ein Verfahren zum Entwurf kombinatorischer Bibliotheken vorgestellt. Es ist sowohl in der Phase der Leitstrukturidentifizierung als auch in der Phase der Leitstrukturoptimierung einsetzbar. Während der Leitstrukturfindung kann *LoFT* verwendet werden, um möglichst diverse Bibliotheken zu generieren, deren Produkte zudem physikochemische Eigenschaftsprofile besitzen, die den Projektanforderungen entsprechen. Während der Leitstrukturoptimierung kann das Verfahren verwendet werden, um fokussierte Bibliotheken zu generieren, deren Produkte ähnlich zu einem oder mehreren bekannten aktiven Molekülen beziehungsweise unähnlich zu unerwünschten Molekülen sind. Auch hier ist es möglich, auf ein gewünschtes Eigenschaftsprofil zu fokussieren und eine möglichst diverse Auswahl der Reagenzien zu erhalten. Die Bewertung der Produkteigenschaften auf Ebene der Reaktanten führt dabei zu Laufzeiten, die eine iterative und nahezu interaktive Anwendung des Programmes zulassen. Im Folgenden wird ein Überblick über die Hauptmerkmale von *LoFT* gegeben. Anschließend werden die Limitierungen dargestellt und Möglichkeiten aufgezeigt, wie das Programm erweitert und weiterentwickelt werden kann.

7.1 Überblick

- *LoFT* bietet mehrere stochastische Algorithmen für die Auswahl von kombinatorischen Subbibliotheken. Das Programm erlaubt die simultane Optimierung unterschiedlicher Kriterien. Durch die Gewichtung der einzelnen Kriterien in der Bewertungsfunktion lassen sich individuelle Entwurfsziele verfolgen. Zudem kann

die Optimierung einer Subbibliothek iterativ durch Hinzufügen weiterer Kriterien verfeinert werden.

- Neben dem Design von Subbibliotheken ist auch die Suche nach den besten Produkten bezüglich der Bewertungsfunktion möglich (Cherry-Picking).

- Der Ähnlichkeitsvergleich zwischen Feature-Trees wird verwendet, um unterschiedliche Designziele zu realisieren. Zum einen können Produkte ähnlich zu bekannten biologisch aktiven Molekülen generiert werden. Zum anderen ist es möglich, Substanzen zu selektieren, die möglichst unähnlich zu unerwünschten Molekülen sind. Der Feature-Tree-Vergleich kann des Weiteren zur Diversifizierung der Reagenzienauswahl und somit der entstehenden Produkte verwendet werden.

- Eine Vielzahl an physikochemischen Eigenschaften ist verfügbar, um Produkte und Fragmente zu filtern und zu bewerten. Auch das Einlesen benutzerdefinierter Eigenschaften und Clusterzuordnungen ist möglich.

- Die Neuartigkeit von *LoFT* besteht darin, dass die Bibliotheken anhand der Produkteigenschaften optimiert werden, ohne die Produkte dafür explizit zusammenzubauen. Die Bewertung der Produkte geschieht auf Basis der jeweiligen Reaktanten durch die Nutzung der Feature-Trees-Technologie und additiver Eigenschaften.

- Durch die Verwendung von Filtermechanismen werden Bibliotheken generiert, deren Produkte ein gewünschtes Eigenschaftsprofil besitzen. Zudem beschleunigen die Filter den Programmablauf, da lediglich Feature-Tree-Vergleiche durchgeführt werden müssen, wenn das betrachtete Produkt die Filterkriterien erfüllt. Zudem erlaubt es die Integration logischer Verknüpfungen in die Filtersprache, Fragmente unterschiedlicher Linktypen nach verschiedenen Kriterien zu filtern. Wurden die Filterkriterien jedoch zu strikt gewählt, entsprechen keine oder nur wenige Produkte diesen Kriterien. In diesem Fall führt die Berücksichtigung der Eigenschaften in der Bewertungsfunktion zu Bibliotheken, bei denen ein Großteil der Produkte den Eigenschaften entspricht.

- *LoFT* verfügt über mehrere Mechanismen, um die Bibliotheken sowohl intern als auch untereinander zu diversifizieren. In der Validierung wurde gezeigt, dass die

Verwendung von Cluster-IDs zu einer diversen Reagenzienauswahl führt und in Verbindung mit anderen Optimierungskriterien eingesetzt werden kann.

- Durch die Sortierung der Reagenzien anhand ihrer Ähnlichkeit zu einem Teil der Anfrage ist oftmals eine effiziente Vorauswahl möglich. Diese ist ein möglicher Ausgangspunkt für die weitere Optimierung. Auch die Auswahl der ähnlichsten Produkte (Cherry-Picking) ist dadurch effizient durchführbar. Im Gegensatz zu FTrees-FS, welches die Produkte ausschließlich anhand ihrer Ähnlichkeit zur Anfrage selektiert, enthält die Lösungsliste die ähnlichsten Produkte, die zudem die gewünschten physikochemischen Eigenschaften besitzen.

- *LoFT* basierte zunächst auf der Flex*-Bibliothek und verwendet mittlerweile die gemeinsam entwickelte *NAOMI*-Bibliothek. *NAOMI* beschleunigt das Einlesen einer kombinatorischen Bibliothek um Faktor 22-25. Zudem wird der Speicherbedarf mehr als halbiert. Dadurch wird die Verwendung von größeren Fragmenträumen erst möglich. Des Weiteren erlaubt das Chemiemodell von *NAOMI* die chemische Validierung der Moleküle, Fragmente und der Verknüpfungsregeln von Fragmenträumen. Es führt zu einer eindeutigen und konsistenten internen Repräsentation der Fragmente und Moleküle unabhängig vom Eingabeformat. Die Generierung der Feature-Trees unter Verwendung des Chemie-Modells hat unter anderem drei Vorteile: Erstens sind die Feature-Trees konsistent zu den weiteren im Programm verwendeten Deskriptoren. Zweitens ist die Generierung so effizient, dass die Deskriptoren nur bei Bedarf im Programm generiert werden. Dadurch sind die Feature-Trees immer konsistent zu der internen Molekülrepräsentation. Nicht zuletzt führt die neue Annotation zu einer leichten Verbesserung der Anreicherung bei den untersuchten Datensätzen.

- Der Feature-Tree-Vergleich wurde so erweitert, dass regioisomere Strukturen unterschieden werden können. Dafür wurde ein neuer Feature-Tree-Generierungsmodus eingeführt, der allen terminalen Schweratomen einen eigenen Knoten zuweist. Dadurch werden beim Entwurf von fokussierten Bibliotheken die Fragmente ausgewählt, deren aromatische Ringe die gleichen oder zumindest ähnliche Substitutionsmuster wie die aromatischen Ringe des Anfragemoleküls aufweisen. Wie zu erwarten, führt die Erweiterung bei Anreicherungsexperimenten nur dann zur Verbesserung, wenn die aktiven Substanzen ähnliche Substitutionsmuster aufweisen.

- Es wurde eine Möglichkeit geschaffen, die Feature-Tree-Matchings so einzuschränken, dass Bibliotheken mit bestimmten Zuordnungen generiert werden können. Dadurch kann verhindert werden, dass eine Subbibliothek mehrere alternative Matchings aufweist. Durch die 2D-Visualisierung der einander zugeordneten Atome ist eine einfache Validierung durch den Nutzer möglich.

- Es wurde eine Schnittstelle zu FlexS [194] integriert, um in einem Vor- beziehungsweise Nachbearbeitungsschritt die Reagenzien und Produkte durch die 3D-Überlagerung mit der Anfrage filtern zu können. Als Startpunkt für eine effiziente 3D-Überlagerung dient die Feature-Tree-Zuordnung der Baugruppen.

- Die Laufzeit des Programmes hängt von der Anzahl der Anfragemoleküle, sowie von der Größe des Anfrage- und Grundgerüst-FTrees ab. Der neue FTree-Generierungsmodus erlaubt die Unterscheidung von regioisomeren Substrukturen, führt jedoch zu einer erhöhten Laufzeit. Die Laufzeit ist zudem linear abhängig von der Größe der zu entwerfenden Subbibliotheken. Bei der Generierung kleiner fokussierter Bibliotheken ist das Programm interaktiv verwendbar.

7.2 Limitierung

Das vorgestellte Verfahren verwendet ein explizites Grundgerüst. Durch die Beschränkung auf Reagenzien mit genau einem Linkatom, ist es möglich den Ähnlichkeitsvergleich zu beschleunigen, wie in Kapitel 5.12 beschrieben. Die Anwendung ist beschränkt auf Fragmenträume, bei denen durch die Regeln nur azyklische Bindungen geknüpft werden.

7.3 Mögliche Erweiterungen

Auch nach Abschluss dieser Arbeit bieten sich viele Erweiterungsmöglichkeiten. Einige Ideen sollen nachfolgend aufgezeigt werden:

- Um *LoFT* einer größeren Gruppe von Anwendern zugänglich zu machen, ist eine graphische Benutzeroberfläche notwendig. *LoFT* wurde bereits so angelegt, dass eine sukzessive Erweiterung mit graphischen Elementen möglich ist (siehe Anhang

C). Zum besseren Vergleich von fokussierten Bibliotheken könnten beispielsweise Histogramme und Streudiagramme der Eigenschaftswerte ausgegeben werden.

- Die Etablierung einer Datenbank zur Speicherung der Fragmente würde einige Vorteile mit sich bringen. So könnten die Fragmente effizient selektiert und gefiltert werden, ohne dass die Fragmente zuvor in den Speicher geladen werden müssen. Da die Fragmente bei *LoFT* erst für das Schreiben der Resultate benötigt werden, könnten nur die jeweiligen Deskriptoren geladen werden. Eine solche Datenbank würde das Verwalten von Reagenzien, Reaktions-Schemata und Grundgerüsten erleichtern.

- Weitere Ähnlichkeitsmaße und Deskriptoren sind ebenfalls von Interesse, zum Beispiel Fingerabdrücke, die die atomare Umgebung kodieren [73, 74] wie *Extended Connectivity Fingerprints (ECFP, FCFP)* [75, 76], oder der maximale gemeinsame Subgraph (MCS, siehe zum Beispiel Raymond et al. [217, 218]). Diese Deskriptoren können insbesondere zur Clusterung der Reagenzien verwendet werden.

- Zusätzliche Clusterverfahren wie k-medoid [122], Jarvis-Patrick [123] und das Verfahren von Ward [120] könnten integriert werden. Interessant wäre auch, ein dichtebasiertes Verfahren wie zum Beispiel OPTICS [219] zu verwenden. Für die Anwendung auf großen Datenmengen sind speicherrestringierte Clusteralgorithmen wie MC-UPGMA [220] eine Option.

- Über 90% der Laufzeit von *LoFT* fließt in die Feature-Tree-Vergleiche. Eine deutliche Verbesserung würde sich durch die Überarbeitung des Feature-Tree Codes erzielen lassen, so dass die nebenläufige Berechnung von Vergleichen möglich ist.

- Um den FTree-Deskriptor weiterzuentwickeln, könnten de Unterschiede zwischen alter und neuer Feature-Tree Beschreibung genauer untersucht werden. Unter Verwendung des *NAOMI*-Modells könnten an den Knoten zusätzliche Eigenschaften wie zum Beispiel die Löslichkeit annotiert werden.

- Zwar existiert eine Anbindung an FlexS [194], dennoch ist die direkte Integration von 3D-Deskriptoren in *LoFT* von Interesse, insbesondere zur Bewertung während der Optimierung. Ist die Bindetasche des Proteins bekannt, könnte das Grundgerüst mit einem Dockingalgorithmus wie FlexNovo [34] platziert werden. Die

Reagenzien könnten so jeweils einzeln bewertet und entsprechend gefiltert werden.

A

Anreicherungsdiagramme

In den folgenden Abschnitten sind die Anreicherungsdiagramme für die einzelnen Aktivitätsklassen des aufbereiteten Hert-Datensatzes [166, 203] (siehe Anhang A.1) als auch des DUD-Datensatzes [202] (siehe Anhang A.2) aufgeführt.

A.1 Hert-Datensatz

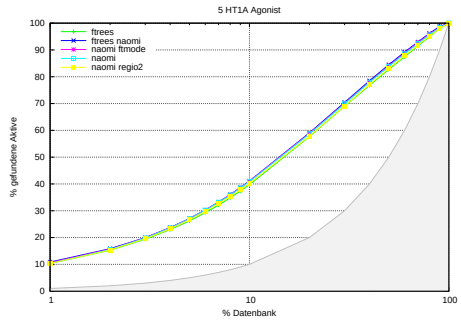
Protein	Anzahl	Anzahl der herausgefilterten Liganden
5HT1A Agonisten	849	5
5HT Reuptake Inhibitoren	359	1
Cyclooxygenase Inhibitoren	635	0
Substance P Antagonisten	1246	72
5HT3 Antagonisten	742	0
D2 Antagonisten	416	1
HIV Protease Inhibitoren	805	185
Renin Inhibitoren	1139	541
Thrombin Inhibitoren	811	51
Protein Kinase C	439	20
Angiotensin II AT1 Antagonisten	2051	50
Inaktive	92958	7258

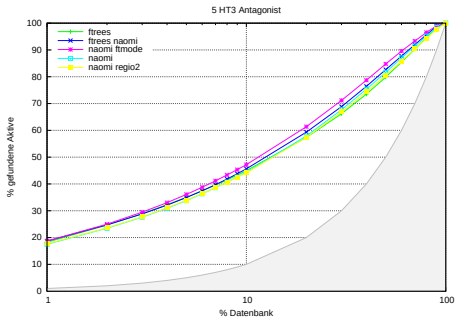
Tabelle A.1: Einteilung des Hert-Datensatzes. Die dritte Spalte listet die Anzahl der Moleküle mit einem Molekulargewicht von mehr als 700 Dalton. Diese wurden aus dem Datensatz entfernt.

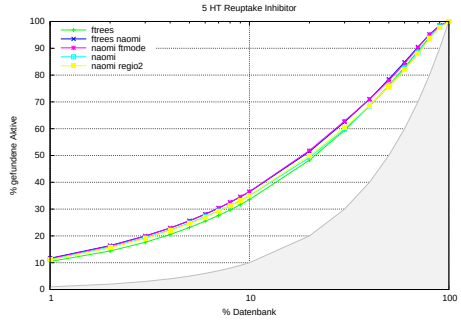
Dieser Abschnitt listet die Anreicherungsdiagramme für die einzelnen Aktivitätsklassen des Hert-Datensatzes [203] (siehe Tabelle A.1). Da einige aktive Substanzen nicht korrekt zugeordnet sind, wurde der Datensatz erstellt, wie in [166] beschrieben. Es wurde jedoch ein aktualisierter MDDR-Datensatz [204] von 2008 verwendet. Zu beachten ist, dass einige Moleküle mehreren Aktivitätsklassen zugeordnet sind und einige Moleküle anhand ihrer ID im MDDR-Datensatz [204] von 2008 nicht mehr gefunden werden konnten. Des Weiteren sind 43 Moleküle des Inaktiven-Datensatzes mit *NAOMI* nicht initialisierbar, da die Beschreibung fehlerhaft ist. Da der Datensatz neben den Liganden auch Strukturen mit einem Molekulargewicht von bis zu 9000 Dalton enthält, wurde der Datensatz mit dem in Abbildung A.1 dargestellten Skript aufgereinigt. Nur die Original-Einträge der Moleküle mit einem Molekulargewicht von bis zu 700 Dalton wurden wieder in eine Datei geschrieben. Das Skript steuert die *NAOMI*-Bibliothek an, die über eine Python-Anbindung mittels SWIG [221] verfügt.

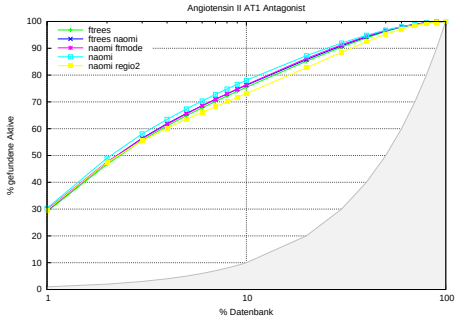
```
import sys
import os
import Naomi

mf = Naomi.MoleculeFactory()

nof_mols = mf.addFile(\"eingabe.sdf\")

of = open(\"ausgabe.sdf\", \"w\")
for i in range (0, nof_mols):
  # Baue Molekuel
  mol = mf.getMolecule(i);
  # Berechne das Molekulargewicht
  mw = Naomi.moleculeCalcMolecularWeight(mol)
  # Filtere
  if (mw <= 700):
    # Ok, schreibe den unveraenderten Eintrag heraus
    of.write(mf.getVerbatimEntry(i))
of.close()
```

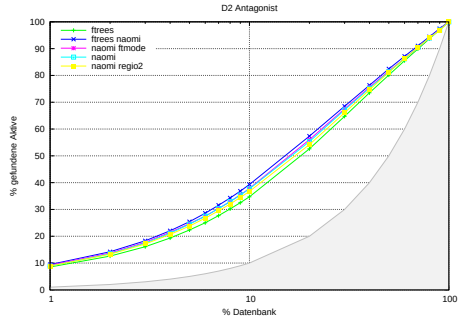
Abbildung A.1: Python-Skript zum Filtern von Datensätzen mit *NAOMI*. Der Übersichtlichkeit halber wurde das Skript gekürzt und modifiziert. Im Original werden Kommandozeilenparameter verwendet und validiert. Zudem wird eine kleine Statistik ausgegeben.

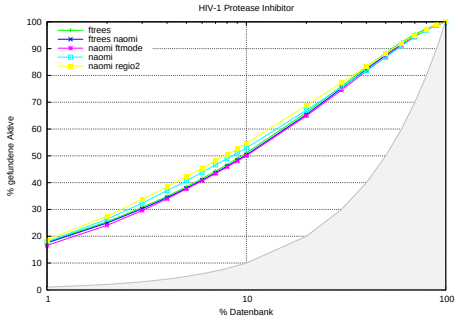

5 HT1A Agonist
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

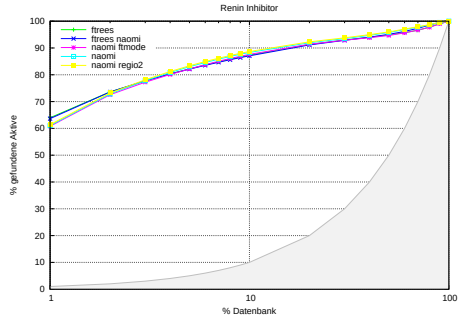

5 HT3 Antagonist
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

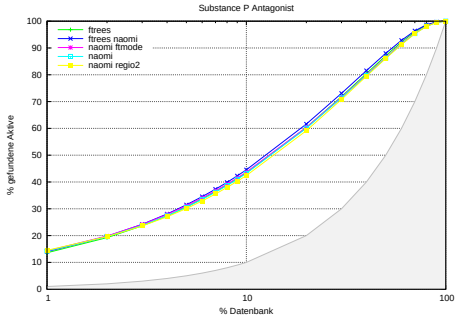

5 HT Reuptake Inhibitor
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

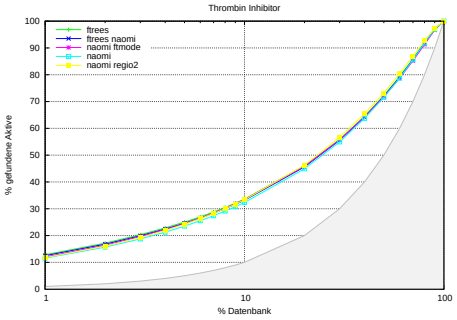

Angiotensin II AT1 Antagonist
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank


D2 Antagonist
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank


HIV-1 Protease Inhibitor
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank


Renin Inhibitor
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank


Substance P Antagonist
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank



A.2 DUD-Datensatz

Dieser Abschnitt listet die Anreicherungsdiagramme für die einzelnen Aktivitätsklassen des DUD-Datensatzes [202] (siehe Tabelle A.2).

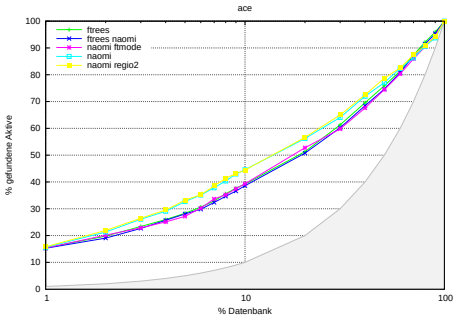
Protein	Aktive	Inaktive
AR	79	2854
ER Agonisten	67	2570
ER Antagonisten	39	1448
GR	78	2947
MR	15	636
PPARγ	85	3127
PR	27	1041
RXRα	20	750
CDK2	72	2074
EGFr	475	15996
FGFr1	120	4550
HSP90	37	979
P38	454	9141
PDGFrb	170	5980
SRC	159	6319
TK	22	891
VEGFr2	88	2906
FXa	146	5745
Thrombin	72	2456

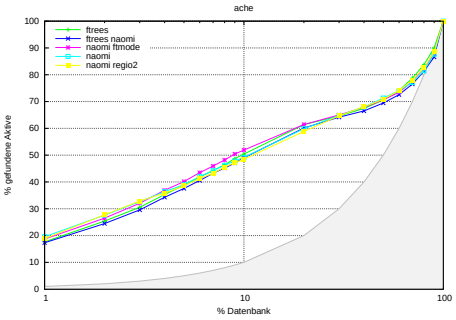
Tabelle A.2: Einteilung des DUD-Datensatzes

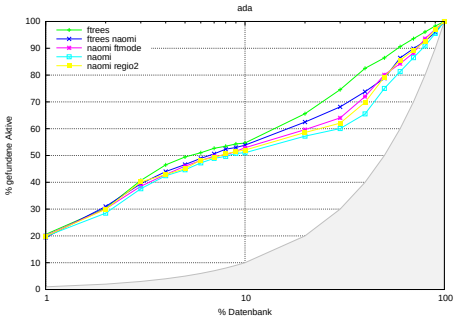
Protein	Aktive	Inaktive
Trypsin	49	1664
ACE	49	1797
ADA	39	927
COMT	11	468
PDE5	88	1978
DHFR	410	8367
GART	40	879
AChE	107	3892
ALR2	26	995
AmpC	21	786
COX-1	25	911
COX-2	426	13289
GPB	52	2140
HIVPR	62	2038
HIVRT	43	1519
HMGR	35	1480
InhA	86	3266
NA	49	1874
PARP	35	1351
PNP	50	1036
SAGG	33	134

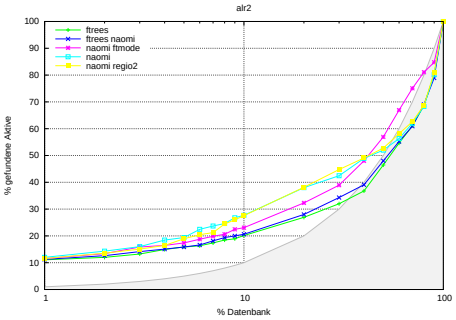
Tabelle A.2: - weitergeführt - Einteilung des DUD-Datensatzes

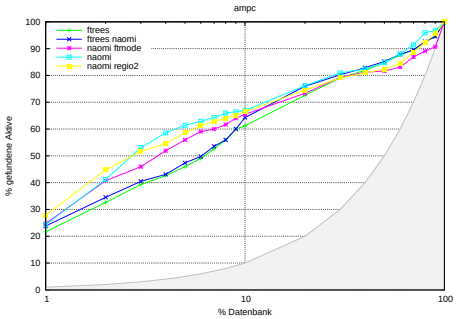
A. ANREICHERUNGSDIAGRAMME

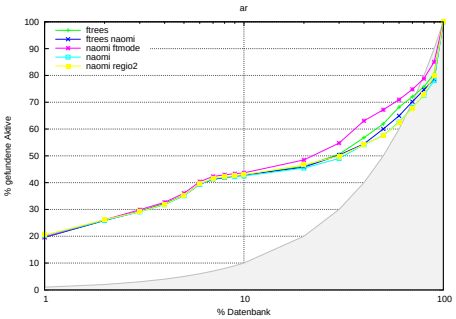


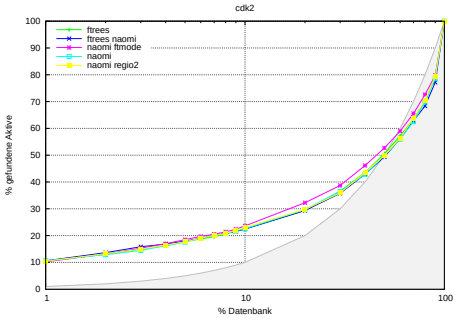


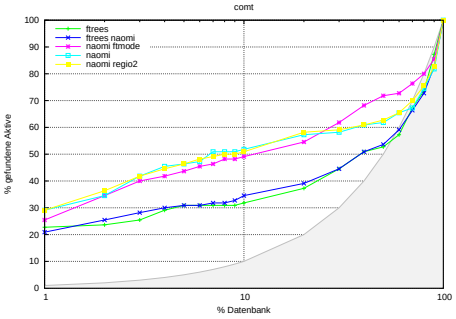


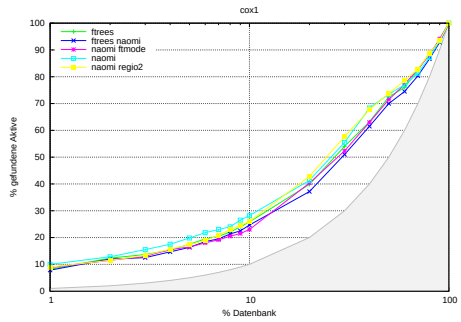


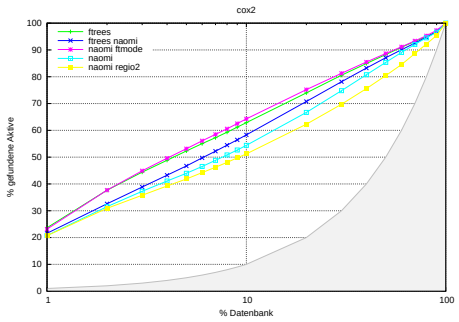


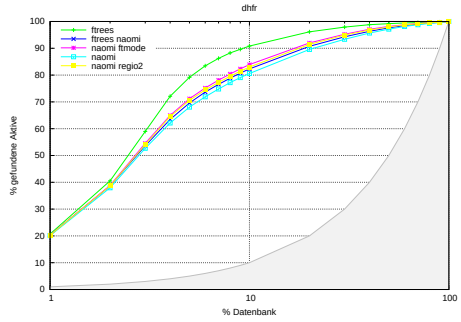


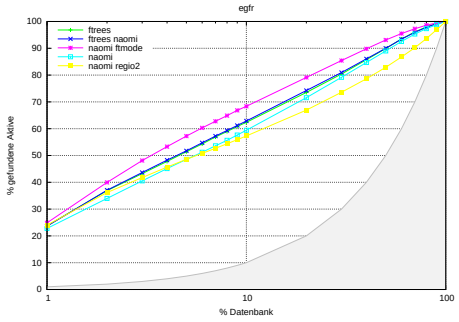


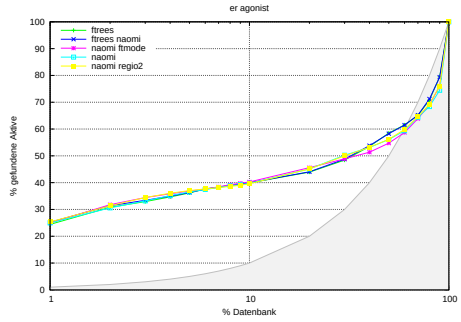


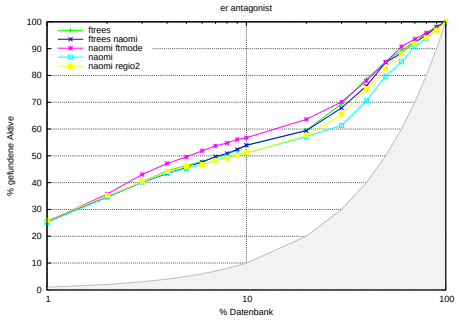

cox1
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

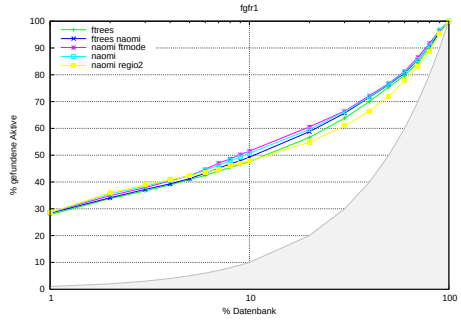

cox2
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

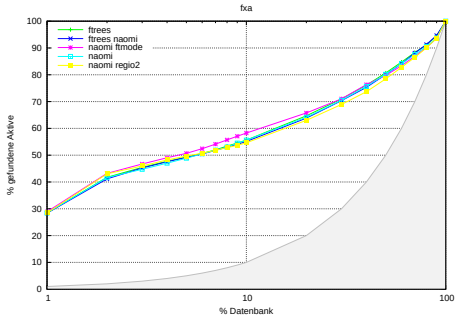

dhfr
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank


egfr
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

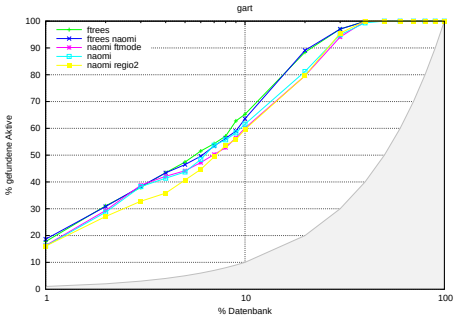

er agonist
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

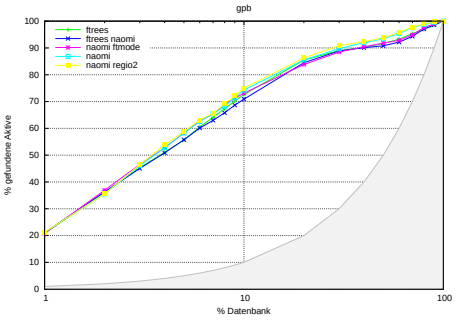

er antagonist
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

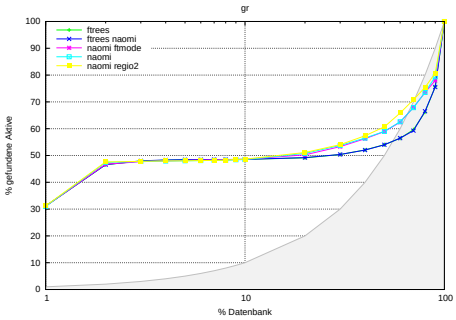

fgfr1
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

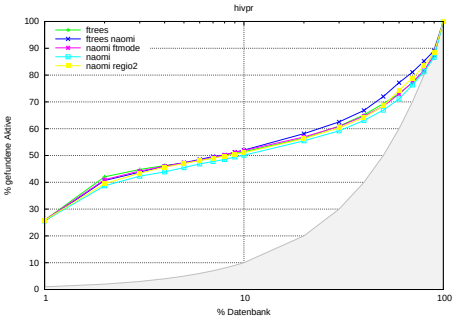

fxa
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

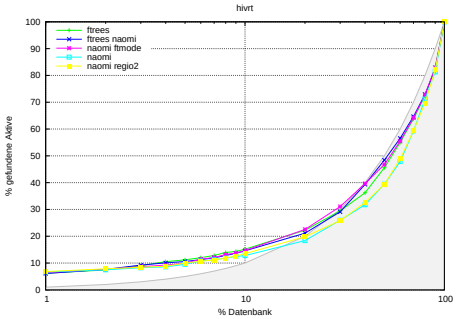
A. ANREICHERUNGSDIAGRAMME

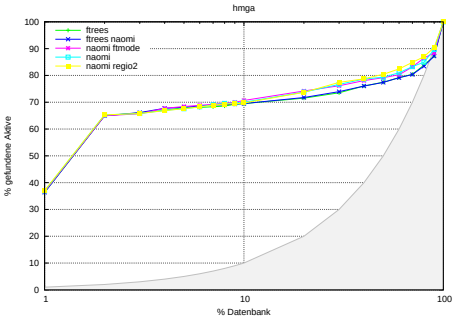


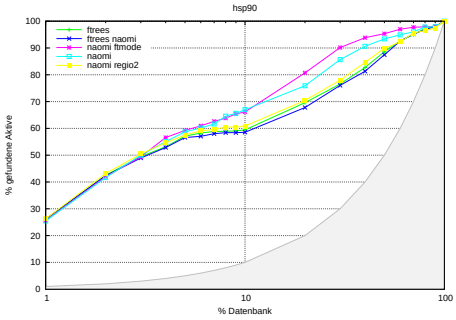


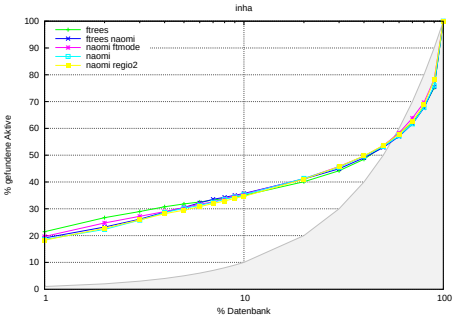


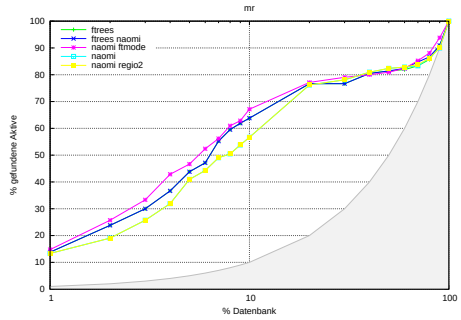


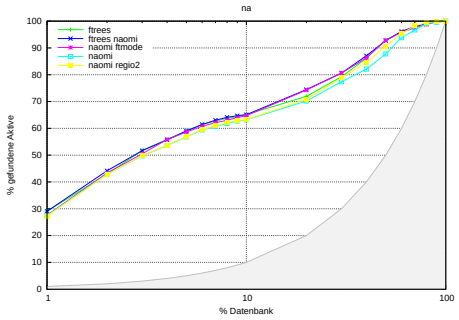


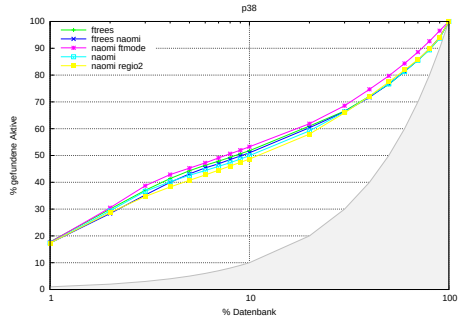


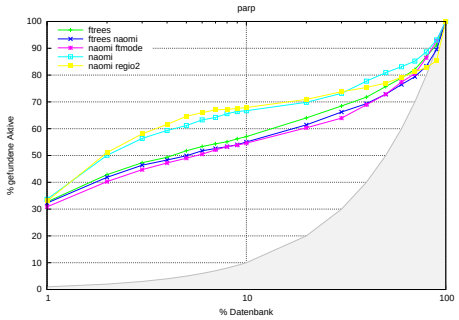


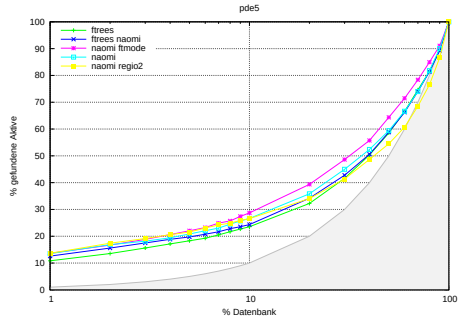


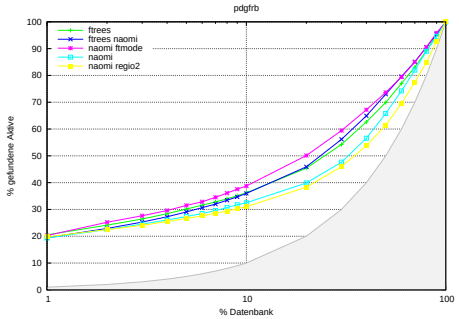

mr
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

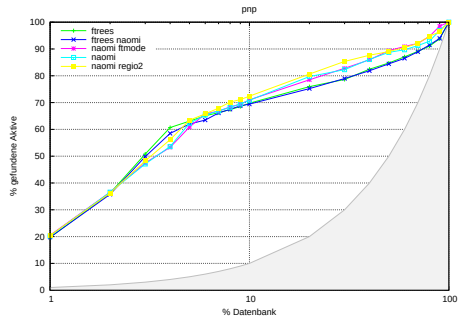

na
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

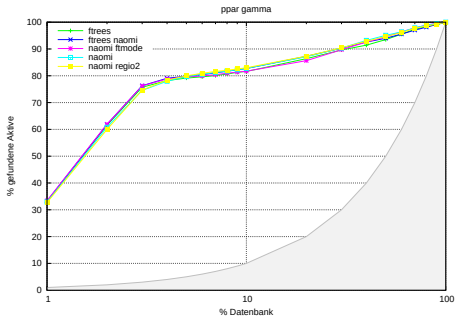

p38
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank


parp
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

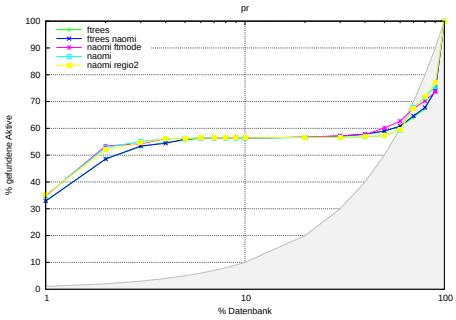

pde5
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

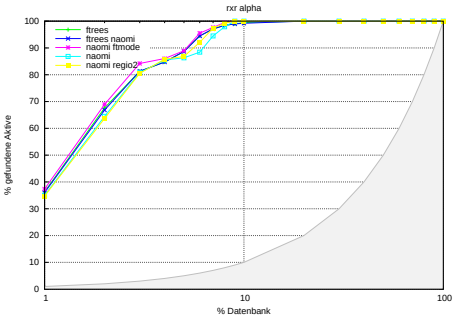

pdgfrb
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

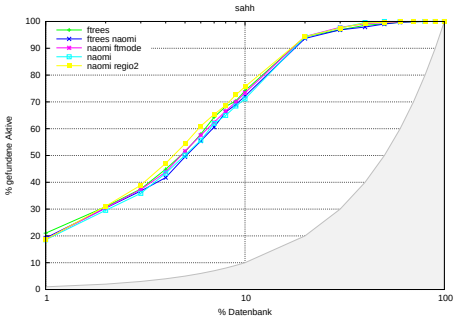

pnp
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

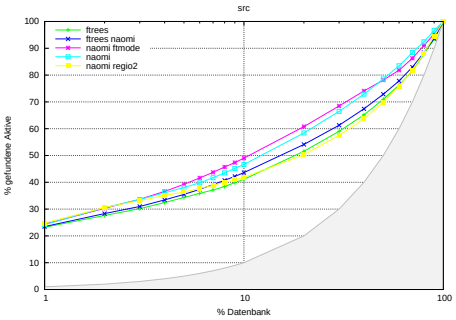

ppar gamma
ftrees
ftrees naomi
naomi ftmode
naomi
naomi regio2
% gefundene Aktive
% Datenbank

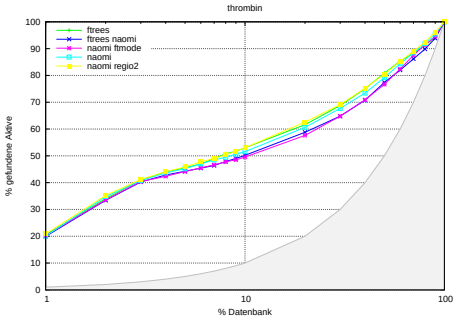
A. ANREICHERUNGSDIAGRAMME

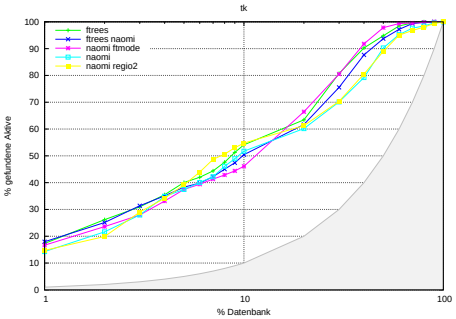


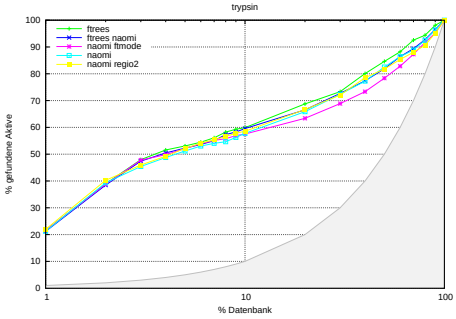


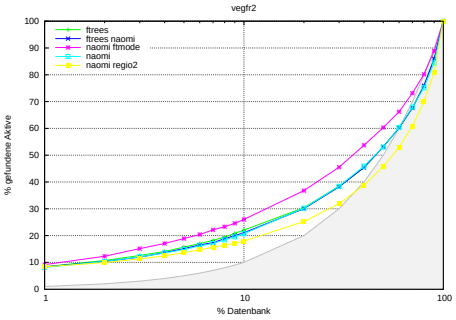












B

Unterstützte Dateiformate

Nachfolgend sind die verwendeten Dateiformate aufgeführt. Anhang B.1 beschreibt die Eingabeformate, Anhang B.2 die Ausgabeformate.

B.1 Eingabeformate

Das obligatorische Eingabeformat für *LoFT* sind Fragmenträume, die die Syntheseprotokolle und Reaktionspartner der kombinatorischen Chemie als Regeln und Fragmente kodieren. Eine Definition des Fragmentraum-Formates findet sich im Handbuch von Colibri [89, 190], welches zur Generierung von Fragmenträumen verwendet werden kann. Neben dem Fragmentraum-Format gibt es weitere, optionale Eingabeformate, die in Tabelle B.1 beschrieben werden. Nachstehend sind explizit die programmeigenen Eingabeformate aufgeführt, die es dem Nutzer ermöglichen, zusätzliche Daten für die Fragmente einzulesen. Aus diesem Grund sind die Formate möglichst einfach gehalten und von Hand editierbar.

So können benutzerdefinierte, additive Eigenschaften für die Fragmente eingelesen und über deren Namen zugeordnet werden. Dafür müssen die Namen allerdings eindeutig sein. Im Format ist die Anzahl der Eigenschaften sowie deren Bezeichner und der Datentyp (0 für ganze Zahlen, 1 für Fließkommazahlen) angegeben. Nach dem Schlüsselwort "@values" werden zeilenweise die Fragmentnamen sowie die Eigenschaftswerte aufgeführt (siehe Abbildung B.1).

Für das Einlesen von Cluster-IDs besteht das Format lediglich aus dem Namen und der ID des Reagenzes (siehe Abbildung B.2).

Distanzmatrizen werden auf ähnliche Art und Weise kodiert (siehe Abbildung B.3). Durch die Zuordnung über die Namen können die Distanzwerte aus mehrere Dateien

Eingabetyp	Endung	Beschreibung
Fragmentraum	*.fsf	Kodierung der Syntheseprotokolle und Reaktionspartner. Die Fragmentraum-Datei enthält einen Verweis auf die Dateien, in denen die Fragmente gespeichert sind. Diese können im SMILES [173], SDF [174] oder MOL2 [175] Format vorliegen.
Anfragemoleküle	*.mol2, *.sdf, *.smi	Moleküle, zu denen die Produkte der optimierten Bibliotheken ähnlich, bzw unähnlich sein sollen. Diese können ebenfalls im SMILES [173], SDF [174] oder MOL2 [175] Format vorliegen.
Additive Fragmenteigenschaften	*	Additive Eigenschaften können für die Fragmente eingelesen und über den Fragmentnamen zugeordnet werden.
Cluster IDs	*	Cluster IDs können für die Fragmente eingelesen und über den Fragmentnamen zugeordnet werden.
Distanzmatrizen	*	Distanzwerte können für die Reagenzien eingelesen werden, wobei die Matrixeinträge über die Fragmentnamen zugeordnet werden.

Tabelle B.1: Die Eingabeformate von *LoFT* und ihre Beschreibung

```
@nof_properties 2
@property price 1
@property importance 0
@values
CORE1    11    22
CORE2    13.02 24
```

Abbildung B.1: Dateiformat zum Einlesen von additiven Eigenschaften

```
REAG1 1
REAG2 2
```

Abbildung B.2: Dateiformat zum Einlesen von Cluster-IDs

eingelesen werden.

```
REAG1 REAG2 0.2
REAG1 REAG3 0.34
```

Abbildung B.3: Dateiformat zum Einlesen von Distanzmatrizen

B.2 Ausgabeformate

Fokussierte Bibliotheken zu vergleichen und auszuwerten ist eine anspruchsvolle Aufgabe, insbesondere wenn das Grundgerüst mehrere Linkatome mit Reagenzien zu dekorieren hat. Um die Auswertung und Weitergabe der Resultate zu erleichtern, werden mehrere Ausgabeformate angeboten. Die infrage kommenden Formate sind in Tabelle B.2 aufgeführt. Weiterhin wird für jede Lösung eine *LoFT*-Datei (*.ld) geschrieben. Diese beinhaltet die Bewertung der Reagenzien und Produkte, ebenso wie statistische Profile (siehe Kapitel 5.11) und die Parameter, die während der Optimierung verwendet wurden. So lassen sich die Daten reproduzieren und vergleichen. Neben diesen Formaten zur Ausgabe von Resultaten ist es möglich, Distanzmatrizen und Cluster-IDs im Dateiformat, wie in Anhang B.1 beschrieben, auszugeben. Des Weiteren können, wie in Kapitel 5.14 beschrieben, FlexS-Dateien für die Reagenzien generiert werden. Um dem Nutzer die Einschränkung der erlaubten Matchings (siehe Kapitel 5.12.2) zu erleichtern, können Dateien für die Anzeige der Anfragemoleküle und Anfrage-FTrees in FlexV [222] generiert werden.

Nachfolgend ist exemplarisch eine *LoFT*-Datei (*.ld) aufgeführt, die beim Design einer 5x5 Bibliothek für den $5HT_{2A}$ Fragmentraum generiert wurde. Gekürzte Teile sind durch Punkte angedeutet.

```
Cores used:
   CORE:core_5HT2a
 Queries used:
     5HT2a_1 (weight: 1.0000)
 Antiqueries used:
   None
 Settings:
 Output dir:                        /home/fischer/results/
 Scoring:                           arithmetic
 Steps to go:                       200000
 Nof max unsuccessful steps:        50000
 Nof solutions per core:            1
 Seed mode:                         take user defined seed
 Seed:                              1
 FTree reagent size filter:         0.5000 1.5000
  (min*nof_nodes(query) <= x <= max* nof_nodes(query))
```

Dateityp	Endung	Fragmente	Produkte	Beschreibung
Fragmentraum	*.fsf	ja	nein	Ausgabe der fokussierten Bibliothek als Teilraum des Eingaberaumes.
UniqueSMILES [173]	*.smi	ja	ja	Eindeutige Repräsentation des Moleküls als Zeichenkette ohne Koordinaten.
MOL2 [175]	*.mol2	ja	ja	Dateiformat zur Speicherung von Molekülrepräsentationen.
SDF [174]	*.sdf	ja	ja	Dateiformat zur Speicherung von Molekülrepräsentationen.
Molekulare Eigenschaften	*.csv	ja	ja	Molekulare Eigenschaften im Spaltenformat (*Comma-Separated Values*) für die Auswertung zum Beispiel mit Spotfire DecisionSite [223].
2D-Diagramme [191]	*.pdf	ja	ja	2D-Diagramme der Moleküle.
2D-FTree Matchings	*.pdf	nein	ja	2D-Diagramme des Feature-Tree Matchings von Anfragemolekül und Produkt (*.pdf), siehe Abbildung 6.29 für ein Beispiel.
FlexS [194]	*.rif, *.lif	nein	ja	FlexS-Eingabedateien die basierend auf dem Feature-Tree Matching die 3D-Überlagerung von Anfragemolekül und Produkt berechnen. FlexS-Überlagerungen können dadurch als Postfilter Anwendung finden (siehe Kapitel 5.14).

Tabelle B.2: Die Ausgabeformate von *LoFT* und ihre Beschreibung.

```
Product size filter:               0.2000 1.5000
 (min*nof_heavy_atoms(query) <= x <= max* nof_heavy_atoms(query))
FTrees regioselectivity factor:    0.0000
Select start reagents by:          random choice
Max nof reagents for each
    core link to consider:         1000000
Algorithm:                         simulated annealing
Temperature:                       1.00000000
Expon. cooling:                    0.99992000
Scoring function:
 #    Property     |     mode       |  weight  | left frontier |   left   |   right  |  right frontier
 ------------------------------------------------------------------------------------------------------
  1 Charge           score uniformly    0.0000          0              0              0              0
  2 Atoms            score uniformly    0.0000          0              0              0              0
  .
  .
  .
 24 FtSim            add score          1.0000          0.0000 <=      Sim to query         <=          1.0000
 25 FtDis            add (1 - score)    0.0000          0.0000 <=    Dissim to antiquery    <=          1.0000

Sorting function:
 #    Property     |     mode       |  weight  | left frontier |   left   |   right  |  right frontier
 ------------------------------------------------------------------------------------------------------
  1 Charge           score uniformly    0.0000          0              0              0              0
  2 Atoms            score uniformly    0.0000          0              0              0              0
  .
  .
  .
 24 FtSim            add score          0.0000          0.0000 <=      Sim to query         <=          1.0000
 25 FtDis            add (1 - score)    0.0000          0.0000 <=    Dissim to antiquery    <=          1.0000

Diversity measurements:
 Diversity measurements for reagents of the same core link:
 Note, that a distance value of 0.1 means a maximum similarity of 0.9
 ---------------------------------------------------------------------------
    Penalty function:
       B----C       max penalty(<= 1.0)
      /      \        % penalty
   __A        D__   0.0

  01 cluster ids: disabled.
  02 average distance: disabled.
  03 pairwise distance: disabled.
Starting reagents were chosen randomly

Filters used:
 Core     filter:
  None
 Reagent filter:
  None
 Product filter:
  None
```

```
Focused library no 1 with score 0.8724, seen after 139801 steps
 Preselected reagents:
  Link R3: 0 reagents preselected
  Link R2: 0 reagents preselected

Reagent profile for core link R3 (1):
            min          max          mean         std. dev.    1th Q(25%)   upper median 3rd Q(75%)
   Charge     0            0            0.0000       0.0000       0            0            0
    .
    .
    .
   Volume    21.7573      82.3359      61.4350      29.2190      38.4101      82.3359      82.3359

Reagent profile for core link R2 (2):
            min          max          mean         std. dev.    1th Q(25%)   upper median 3rd Q(75%)
   Charge     0            0            0.0000       0.0000       0            0            0
    .
    .
    .
   Volume   106.6404     139.9460     126.1540      14.1116     120.9444     123.2932     139.9460

Unified reagent profile for all core links:
            min          max          mean         std. dev.    1th Q(25%)   upper median 3rd Q(75%)
   Charge     0            0            0.0000       0.0000       0            0            0
    .
    .
    .
   Volume    21.7573     139.9460      93.7945      40.3910      38.4101     106.6404     123.2932

Product profile:
            min          max          mean         std. dev.    1th Q(25%)   upper median 3rd Q(75%)
   Charge     0            0            0.0000       0.0000       0            0            0
    .
    .
    .
   Volume   262.7901     356.6744     321.9815      29.6211     296.0958     323.3687     340.0216

Reagent scores:
Core link
   Reagent name (Link, ID, ClusterID) > Score (score without div pen,
                                               #frags from cluster,
                                                          min avg distance,
                                                               min pairwise distance)
 R3
     0xcff17d5c (R4        1660        -) > 0.8803 (0.8803     -      -      - )
     0xf719bae2 (R4        1972        -) > 0.8740 (0.8740     -      -      - )
     0xa046355a (R4        1272      743) > 0.8692 (0.8692     -      -      - )
     0x6344997a (R4         815      743) > 0.8692 (0.8692     -      -      - )
     0x58cef15a (R4         725      743) > 0.8692 (0.8692     -      -      - )
 R2
     0xde3709e4 (R1       21705     1855) > 0.8940 (0.8940     -      -      - )
     0xd1a419c2 (R1       20618     1855) > 0.8715 (0.8715     -      -      - )
```

```
    0x5cf889fa (R1       10207     1855) > 0.8706 (0.8706     -      -      - )
    0x52daca37 (R1        9301     1544) > 0.8654 (0.8654     -      -      - )
    0x97eed1a6 (R1       15435     1710) > 0.8603 (0.8603     -      -      - )


Molecule scores:
   CORE:core_5HT2a_0xcff17d5c_0xde3709e4         > 0.9030
   CORE:core_5HT2a_0xf719bae2_0xde3709e4         > 0.8965
     .
     .
   CORE:core_5HT2a_0x6344997a_0x97eed1a6         > 0.8574
   CORE:core_5HT2a_0x58cef15a_0x97eed1a6         > 0.8574
```

C

Implementierung

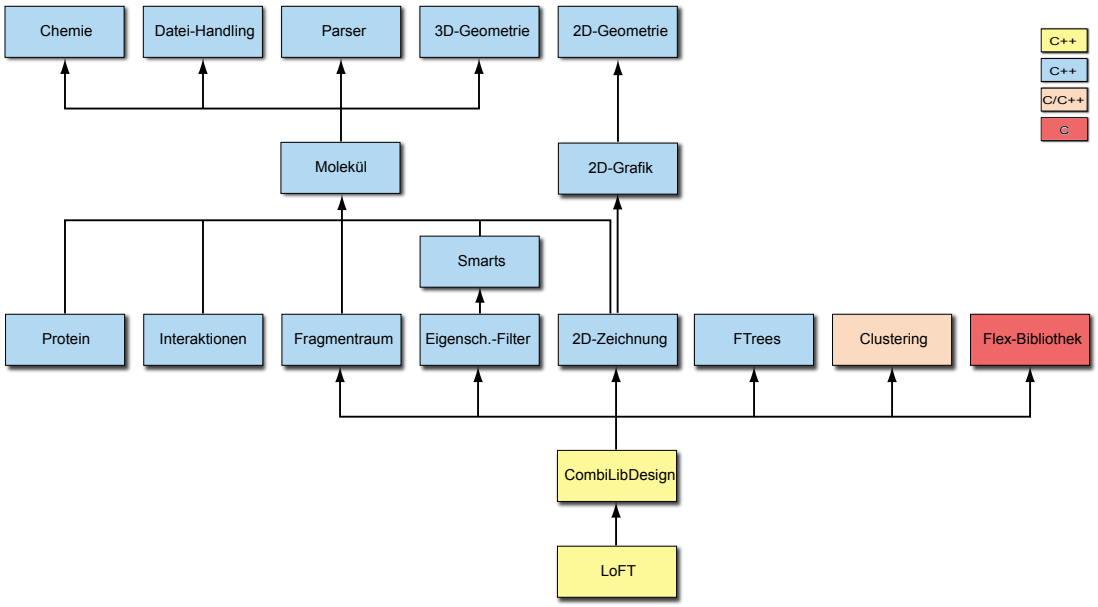


Abbildung C.1: Die Abbildung zeigt den reduzierten Abhängigkeitsgraphen. Kanten von Modulen, die von anderen Modulen bereits eingebunden (zum Beispiel Molekül vom Fragmentraum) werden, sind nicht angezeigt. Es bestehen keine zyklischen Abhängigkeiten. Die *LoFT*-Module (gelb) hängen von Modulen der *NAOMI*-Bibliothek (blau) und dem Clustering-Modul (rosa) ab. Aufgrund der Feature-Tree-Vergleichsalgorithmen und des Simulationsmoduls besteht zudem eine Abhängigkeit von der Flex-Bibliothek (rot).

Der erste Prototyp von *LoFT* [32] wurde zunächst, ebenso wie die zugrundeliegende Flex*-Bibliothek, vollständig in ANSI-C geschrieben. Insbesondere das Fragmentraum-Modul [34], Feature-Trees [30], sowie die Eigenschaftsfilter von FragEnum [68]

und das Clustering-Modul wurden zur Erstellung dieses ersten Prototyps genutzt. Das Clustering-Modul wurde mit einer Benutzungsschnittstelle ausgestattet und ebenfalls in FTrees [205] integriert. Mit Einführung der *NAOMI*-Bibliothek [35] wurde *LoFT* zu großen Teilen in C++ neu implementiert. In diesem Kapitel soll deshalb zunächst auf den generellen Aufbau von *NAOMI* eingegangen werden (Kapitel C.1). Dabei werden insbesondere die für *LoFT* relevanten Teile und Neuerungen vorgestellt. Anschließend wird der Aufbau von *LoFT* beschrieben (Kapitel C.2).

C.1 *NAOMI*

Die *NAOMI*-Bibliothek stellt chemisch valide Molekülrepräsentationen zur Verfügung (siehe Kapitel 4) und wurde zusammen mit Sascha Urbaczek, Adrian Kolodzik und Tobias Lippert entwickelt. *NAOMI* besteht aus hierarchisch abhängigen Modulen (siehe auch C.1), die jeweils einzelne Aufgaben, wie zum Beispiel das Parsen eines Moleküleintrages, übernehmen. Bei der Entwicklung von *NAOMI* wurden die C++-Bibliotheken STL (Standard Template Library) [224], Boost [225] und Qt [226] verwendet. Der stabile und plattformunabhängige Code vereinfacht die Wartung des eigenen Codes und bietet eine Vielzahl an bereits getesteten Funktionalitäten. So stellt die Qt-Bibliothek, welche vor allem zur Entwicklung von graphischen Benutzeroberflächen (zum Beispiel für LeadIT [227], *NAOMI*-Konverter [35]) vorgesehen ist, unter anderem ein Programmiergerüst zum Testen von Modulen zur Verfügung. Es wird verwendet, um Funktionalität testen und fixieren zu können. Dadurch ist es möglich, Submodule, Strukturen, Klassen und Algorithmen auszutauschen und zu modifizieren, aber dennoch das korrekte Verhalten für übergeordnete Module aufrecht zu erhalten. *NAOMI* enthält mehrere Klassen und Funktionen, die Standard-Arbeitsabläufe abbilden. Sie vereinfachen die Nutzung der Module und erleichtern das Testen derselben. Ein Beispiel ist die Funktion *moleculeFromSmiles*, die ein Molekül aus einem SMILES [228] generiert. Aufbauend auf diesen Modulen wurde der *NAOMI*-Konverter als Kommandozeilen-Programm entwickelt und schließlich mit dem Prototypen einer graphischen Benutzeroberfläche in Qt erweitert.

Die Neuimplementierung des Fragmentraum-Moduls in C++ erfolgte in Zusammenarbeit mit Tobias Lippert. Durch die Verwendung von *NAOMI* können die Fragmente eines Raumes nicht nur chemisch validiert, sondern auch parallel eingelesen werden. Die verringerte Speicheranforderung ermöglicht die Nutzung von größeren Fragmenträumen (siehe Kapitel 6.1). Die Fragmente des Fragmentraumes enthalten einen Zeiger auf die jeweils darunterliegende Molekülstruktur (Kompositum-Entwurfsmuster [229]). Durch Referenzzählung können die Ergebnisse von *LoFT* konsistent gehalten werden, ob-

wohl bereits ein neuer Fragmentraum eingelesen wurde. Dafür wird der *shared_ptr* von Boost [225] verwendet. Außerdem wurden das Fragmentraum-Modul und die Funktionalität zum Filtern und Auswerten von molekularen Eigenschaften voneinander getrennt, so dass auch Molekülдatensätze gefiltert werden können. Zudem wurde das Generieren von Feature-Trees basierend auf dem chemischen Modell von NAOMI im Kontext dieser Arbeit in C++ neu implementiert (siehe Kapitel 4.4). Um die Feature-Tree-Vergleichsalgorithmen verwenden zu können, wurde eine Klasse zur Konvertierung in die alte Feature-Tree-Struktur geschrieben. In LoFT wird deshalb der Feature-Tree über einen Adapter angesteuert, so dass ein Austausch der darunterliegenden Struktur problemlos möglich ist. Auch die Schnittstelle des Clustering-Moduls wurde in C++ neu implementiert. Der von LoFT verwendete 2D-Zeichner wurde von Matthias Hilbig auf der Basis des Codes von Patrick Maaß [191] und das SMARTS-Matching von Christian Ehrlich entwickelt. Des Weiteren besteht die NAOMI-Bibliothek aus Modulen, die separat von LoFT entwickelt wurden, wie zum Beispiel ein Modul für die Handhabung von Proteinen. Zu diesen Modulen existieren bei LoFT keine Abhängigkeiten.

C.2 LoFT

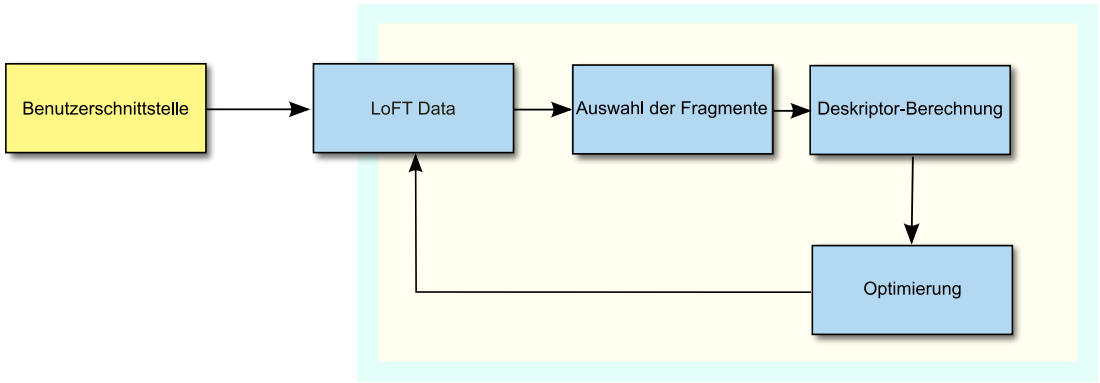


Abbildung C.2: Die Abbildung skizziert den Programmaufbau von LoFT. Von der Nutzerschnittstelle aus kann über ein LoftData-Objekt die Optimierung angesteuert werden.

LoFT kann vom Benutzer generell auf zwei Arten ausgeführt werden. Entweder interaktiv oder automatisch unter Verwendung von Skriptdateien. Wird das Programm interaktiv genutzt, können Dialog-Fenster verwendet werden, um dem Nutzer das Laden von Dateien auf einfache Art und Weise zu ermöglichen. Dies ist notwendig, da die verwendete Flex-Menüführung die Expansion von Dateipfaden nicht unterstützt. Zudem erlaubt es die eingeführte Qt-Anbindung, nach und nach GUI-Fenster für Programm-

teile einzuführen, so dass *LoFT* schrittweise eine graphische Nutzeroberfläche erhalten kann. Trotzdem bleibt *LoFT* auch über Skripte steuerbar, da sich gerade in dem Anwendungskontext eine Vielzahl von Schritten automatisieren und parallelisieren lässt.

Technisch setzt sich *LoFT* aus zwei getrennten Komponenten zusammen. Zum einen besteht es aus dem CombiLibDesign-Modul, welche die Datenstrukturen und Algorithmen enthält. Zum anderen besteht es aus dem *LoFT*-Programm, welches die Benutzerschnittstelle und Menüfunktionen implementiert (siehe Abbildung C.2). Die Abhängigkeiten der beiden Programmteile sind streng hierarchisch, wie der Abhängigkeitsgraph in Abbildung C.1 zeigt. Die nachfolgenden Tabellen C.1 sowie C.2 listen die Submodule und ihre Funktion für das CombiLibDesign-Modul sowie für das *LoFT*-Programm auf.

Submodul/Klasse	Funktion
Forward	Vorwärtsdeklaration der Klassen, Enumeratoren und Typdefinitionen
LoftData	Hauptdatenobjekt, welches die Daten zusammenführt: Es speichert den Fragmentraum, die Anfragemoleküle und die Optimierungsergebnisse.
LoftRun	Speichert die Ergebnisse eines Optimierungslaufes für mehrere Cores. Speichert ebenfalls die verwendete Bewertungsfunktion, die Parameter usw.
Solution	Lösung für ein Grundgerüst, bzw. ein Cherry-Picking
Query	Anfragemolekül
Core	Grundgerüst
Reagent	Reagenz
FocusedLibrary	Fokussierte Bibliothek: Enthält die ausgewählten Reagenzien und ihre Bewertung
FocusedLibraryEnumerator	Enumeration der fokussierten Bibliothek zum Generieren der Produkte oder Bewerten der Bibliothek
CherryPicking	Cherry-Picking-Algorithmus
Scoring	Bewertungsfunktion
DescriptorScoring	Bewertungsfunktion für molekulare Deskriptoren
DesirabilityScoring	C++ Template der Wünschbarkeitsfunktionen [65] für die verschiedenen Datentypen (int, unsigned int, double)

Tabelle C.1: Die wichtigsten Submodule und Klassen des CombiLibDesign-Moduls und ihre Funktion (Namensraum *CombiLibDesign*).

Submodul/Klasse	Funktion
Diversity	Mechanismen zur Berechnung der Diversität zwischen den Reagenzien beziehungsweise fokussierten Bibliotheken
OutputGenerator	Generiert Ausgabedateien für die Lösungen
FlexSWriter	Schreiben von FlexS-Daten
FtreeMatchingWriter	Generiert 2D-Diagramme aus dem FTree-Matching von Molekülen
Ftree	Fassade, die den Zugriff auf den Feature-Tree-Deskriptor erlaubt
FtreeComparison	Feature-Tree-Vergleichsfunktionen im *LoFT*-Kontext
ClusteringAdaptor	Adapter der Funktionen für das Clustern von Reagenzien zur Verfügung stellt
Parameter	Struktur, welche die verwendeten Parameter zusammenfasst
CalculationData	Sammlung temporärer Daten, die während der Optimierung verwendet werden. Beispiel sind die Matrizen für den Feature-Tree-Vergleich.
ld_algorithms	Hilfsfunktionen die z.B. die Fragmente für eine Optimierung auswählen.
ld_simulation	Adapter zur Anbindung an das Modul für die stochastische Optimierung

Tabelle C.1: - weitergeführt - Die wichtigsten Module und Klassen des CombiLibDesign-Moduls und ihre Funktion (Namensraum *CombiLibDesign*).

Submodul/Klasse	Funktion
Loft	Zugriff auf ein LoftData Objekt (Singleton), unterscheidet zwischen GUI/Kommandozeilen-Modus
LoftWidgets	Von *LoFT* verwendete GUI-Fenster
Main	Hauptfunktion des Programmes
FlexInitialization	Aufbau des Menübaumes; Einlesen der Parameter aus der Konfigurationsdatei
Input	Einlesen von benutzerdefinierten Moleküleigenschaften

Tabelle C.2: Die Submodule des *LoFT*-Programmes und ihre Funktion. Die Submodule implementieren die Benutzerschnittstelle (Namensraum *LoFT*).

Submodul/Klasse	Funktion
ScoringUserInterface	Abfragen zum Setzen der Bewertungs- und Diversitätsfunktionen
MenuFocus	Menüfunktionen zur Bibliotheks-Optimierung
MenuQuery	Menüfunktionen zur Auswahl der Anfragemoleküle
MenuFragspace	Menüfunktionen zur Auswahl des Fragmentraumes
MenuFilter	Menüfunktionen zum Filtern von Reagenzien, Grundgerüsten und Produkten
MenuClustering	Menüfunktionen zum Clustern von Reagenzien

Tabelle C.2: - weitergeführt - Die Submodule des *LoFT*-Programmes und ihre Funktion.

Der Zugriff auf das Hauptdatenobjekt des CombiLibDesign-Moduls wurde auf Ebene der Benutzerschnittstelle als Singleton-Entwurfsmuster [229] implementiert. Dies schließt jedoch nicht die Möglichkeit aus, mehrere Instanzen des Hauptdatenobjektes zu verwenden.

D

Einführung in die Benutzungsschnittstelle

Das folgende Tutorium soll dem Benutzer einen einfachen Einstieg in die Benutzung von *LoFT* ermöglichen. Da das Programm eine internationale Anwenderschaft ansprechen soll, ist das Tutorium in Englisch geschrieben.

LoFT

Tutorial

Robert Fischer

Contents

1

About *LoFT*

The design of a new focused library is a complicated task. Typically, several criteria have to be considered simultaneously during compound selection. For example a certain balance between similarity to known actives and diversity between the selected compounds should be achieved.

1.1 General idea of *LoFT*

LoFT[2] is a program for focused combinatorial library design applying multi-objective optimization and uses the feature tree descriptor for similarity/dissimilarity measurement. By applying the comparison directly on fragment level and the use of various physico-chemical properties as well as user defined criteria, *LoFT* is able to design focused libraries efficiently without combining the fragments (see 1.7). Several stochastic algorithms are provided for traversing the search space (see 1.9). In addition to simulated annealing and threshold acceptance, a cherry picking, which selects the n best products from the search space, is available. To select the best reagents, a weighted multi-objective scoring function, filtering rules and diversity mechanisms are applied.

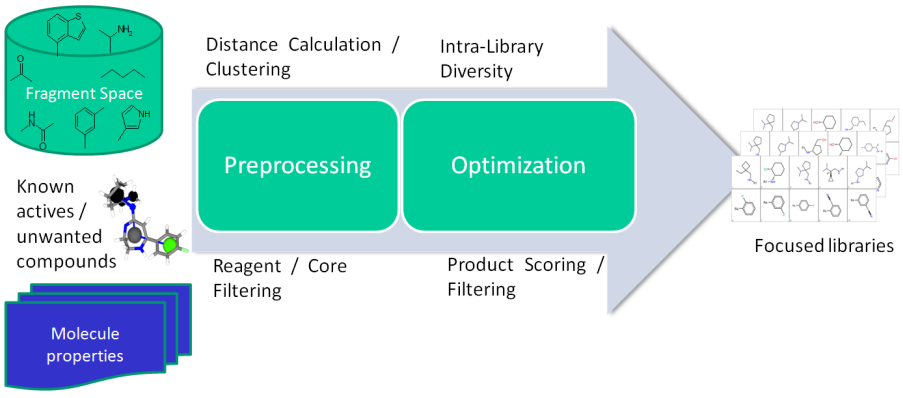


Figure 1.1
This figure depicts the general idea of *LoFT*. From a given fragment space (see 1.4), according to a scoring function and the similarity to one or more query molecules, the best combination of reagents for one or more cores (with the same link patterns) is chosen. A focussed library is built.

1.2 Features

LOFT combines several key features to provide a library design framework. These include:

- Physico-chemical properties (1.6)
- Weighted multi-objective scoring schema
- Optimization algorithms
- Hierarchical clustering
- Diversity mechanisms
- Filters on reagent, core and product level
- Query/antiquery focusing
- Similarity comparison on fragment level

1.3 Limitations

We discriminate between cores (fragments with more than 1 link) and reagents (1 link). The program was designed to generate the best reagent selection for a single (or at least a few) core(s with the same link patterns). Therefore the generated libraries consist always of a certain scaffold (the core). For R_1-R_2 libraries, a dummy core was introduced. R_1-...-R_n libraries are not considered yet. Furthermore using the feature tree descriptor ring closure reactions can be modelled only indirectly.

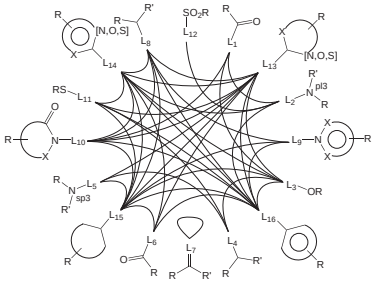
1.4 Requirements and Input

LOFT uses chemical fragment spaces as underlying search space (see figure 1.2). A fragment space mainly consists of a collection of fragments and a set of connection rules specifying which fragments can be combined. Fragments and query molecules can be provided in MOL2, SMILES and SDF format.
The fragment space must be provided in the FlexNovo format, which can be generated using Colibri[1]. For descriptor consistency, all descriptors (including the feature tree descriptor) are generated on the fly within *LOFT*.

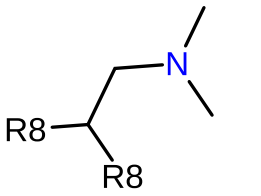
1.5 Output

The designed sublibraries and products can be saved in different formats:

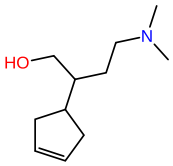
- Fragments space (.fsf, only sublibraries)
- MOL2 (.mol2)
- SMILES (.smi)
- SDF (.sdf)



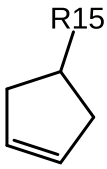
Connection rules



A sample core (2 or more links)



A sample product (or query)



A sample reagent (1 link)

Figure 1.2
This figure depicts a typical fragment space. In the upper left, exemplarily some connection rules are shown. The fragments can be combined to products applying those rules. *LoFT* discriminates between cores and reagents.

- 2D sketches of products (.pdf)
- Molecular properties of products (.csv)
- 2D feature tree mappings of queries and products (.pdf)
- Data for starting a *FlexS* superpositioning (.rif)

Per default, a *LoFT*-file is written (.ld). It contains date and time, the applied parameters, filters and the scores of reagents and products. A CSV file can be generated, which enables the user to inspect the properties of the focused libraries e.g. with Spotfire[6]. 2D drawings support a direct visualisation of products and their feature tree mapping according to the query molecules. Furthermore, it is possible to write the computed distance matrices to file.

1.6 Physico-chemical properties and descriptors

The following descriptors are provided for computation. For every property, a specific desirability scoring function (see figure 1.3) can be defined:

- Charge
- Number of non-hydrogen atoms
- Number of acceptors
- Number of donors
- Number of hetero atoms
- Number of aromatic atoms
- Number of halogens
- Number of inorganic atoms
- Number of non-hydrogen bonds
- Number of rotatable bonds
- Maximum path of contiguous rotatable bonds
- Number of ringsystems
- Number of aromat
- Number of rings
- Number of aromatic rings
- Maximum number of atoms in ringsysstem
- Number of R/S centers
- Number of E/Z bonds
- Number of link atoms
- Molecule weight
- TPSA
- PLogP
- Volume
- Link name
- Smarts
- ve properties

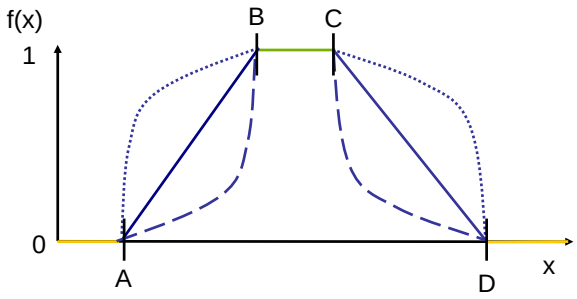


Figure 1.3
By definition of the boundaries a desirability scoring function can be applied for additive properties. It maps the property values in the range [0,1]. For each property, the point A, B, C, and D can be set. A value in the range of [B,C] is desired (score 1), a value less than A or greater than D is unwanted (score 0).

1.7 (Dis)similarity to given queries

LOFT incorporates the feature tree descriptor for similarity comparisons. A feature tree abstracts the molecular graph to a tree structure. The molecule is split at all acyclic non-terminal bonds. Simple cycles are condensed into single nodes and complex cyclic systems are decomposed. These 'building blocks' form the nodes in a feature free. The nodes are only connected if the building blocks are connected. This preserves the overall topology of the molecule. Finally, the physico-chemical properties of the original molecule building blocks (size, #donors, #acceptors, aromaticity, etc.) are assigned to the corresponding nodes (see figure 1.4). For similarity comparison, the Match Search algorithm is used, searching

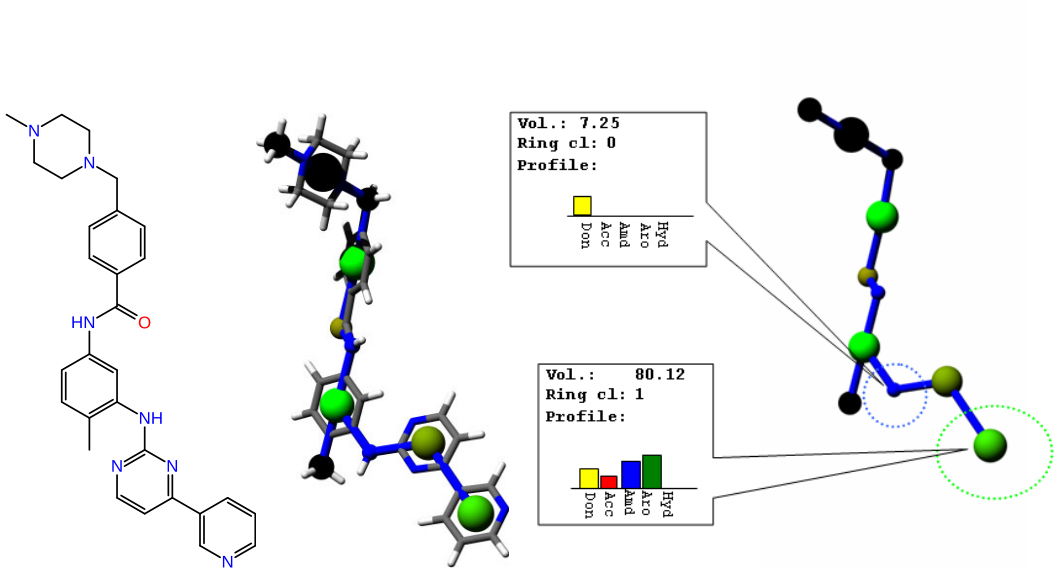


Figure 1.4
Example for Gleevec (a) and its corresponding feature tree (b). The nodes are labeled with physico-chemical properties (c).

for a maximum weighted bipartite matching. Therefore the subtrees are aligned on each other in a hierachical manner, splitting the subtrees recursively in smaller subtrees using a dynamic programming approach. This approach can be easily adapted to the fragment level[4] and to combinatorial library design[2]. A more detailed description of the feature tree comparison procedure can be found in the publications of Rarey et al.[5, 3] Hence the library can be focused by similarity to one or more query molecules. Also, molecules can be used as antiqueries resulting in products different from them. Using more than one query or antiquery, the influence of these molecules can be weighted. Note, that the number of queries affects the run time, because the number of similarity comparisons is increased.

1.8 Library diversity

A single or complete linkage algorithm can be used to cluster the reagents using feature tree (dis)similarity. Diverse libraries can be achieved by restricting the number of reagents from one cluster. In addition the computed distance matrix can be used for an either pairwise or average dissimilarity minimum restriction during focused library design. The distance matrix can be loaded from file if computed once.

1.9 Optimization algorithms

LoFT provides several algorithms for traversing the search space:

- Simulated annealing

- Threshold acceptance
- Great deluge
- Hill climbing
- A modified harmony search
- Enumerator (for at least very small spaces)
- Cherry picking (selecting the n best products according to the scoring function)

2

Getting started - a tutorial introduction to *LoFT*

To facilitate the first usage of *LoFT*, a short tutorial is given, following the main workflow depicted in figure 2.1:

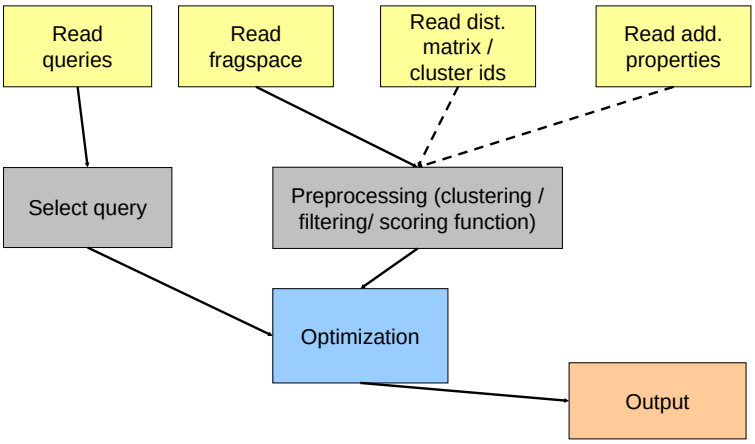


Figure 2.1: Simplified workflow of *LoFT*

After reading input data (yellow boxes), some individual preprocessing steps are performed (grey boxes) and the optimization is started (blue box). The results can be written to file afterwards (amber box).

In the following, we will load data from file, define the design criteria and start an optimization. Afterwards we write the results to file.

```
 __      _____   ____   ____
(  )    (  _  ) ( ___) (_  _)
 )(__    )(_)(   )__)    )(
(____)  (_____) (__)    (__)
```

```
 ZBH - Center for          Focussed molecular library design
 Bioinformatics
 University of Hamburg     Debug-Version:    0.9.5       (27.12.10)
 Bundesstrasse 43          Modules:
 20146 Hamburg
 Germany                   Authors:    Robert Fischer, Matthias Rarey

 BioSolveIT GmbH           Copyright: ZBH, University of Hamburg, Germany
 An der Ziegelei 75                   BioSolveIT GmbH, Sankt Augustin, Germany
 53757 St. Augustin
 Germany
_____________________________________________________________________________


   Running on gera (Linuxx86_64 2.6.27.37-0.1-default) with 2 processors.
 >> LoFT configuration file '/home/goofy/config_loft.dat' loaded.
 >> LoFT license check (BioSolveIT keys): succeeded.
   >> Licensed modules: LoFT [DECRYPT]
```

After the program has started correctly, the prompt occurs.

```
LOFT>
```

All global variables can be changed directly. The default values are set in the configuration or settings file.

```
LOFT> SET  NOF_SOL 3
 >> Data may be inconsistent (RELOAD?).
```

Now we enter the fragment space submenu and read a fragment space from file.

```
LOFT> FRAGSPACE
LOFT/FRAGSPACE> READ
```

If *LoFT* was started in 'GUI' mode, a windows will open (see figure 2.2):
Otherwise enter the file name:

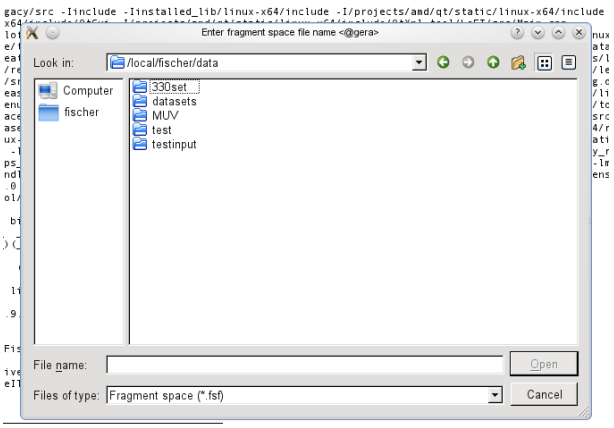


Figure 2.2
Window asking for the fragment space file name.

```
LOFT> FRAGSPACE
LOFT/FRAGSPACE> READ  /home/goofy/data/TestSpace.fsf
   Obtaining all positions in file...
```

```
Fragment space successfully loaded from /home/goofy/data/TestSpace.fsf
Number of fragments: 101
Number of link types: 25
Number of connection rules: 43
Number of possible link connections: 2120

Number of links histogram:
 #Links:           1     2     3     4
 #Occurences:     85    14     2     0

Link type occurence histogram:
 R1              7
 R2              1
 R3              1
 R4              1
 R5              1
 R6              1
 R7              1
 R8              1
 R9              8
 R10             1
 R11             1
 R12             1
 R13             1
 R14             1
 R15             1
 R16             1
 R17            10
 R18            10
 R19             8
 R20             2
 R21             7
 R22            10
 R23             3
 R24            10
 R25            30

Link compatibilities:
 R1:   R15
 R2:   R25
 R3:   R25
 R4:   R22
 R5:   R21  R23  R25
 R6:   R9   R20
 R7:   R22  R24
 R8:   R24
 R9:   R6
 R10:  R22  R24
 R11:  R21  R23  R25
 R12:  R24
 R13:  R24
 R14:  R21  R23  R25
 R15:  R1   R23
 R16:  R21  R23  R25
 R17:  R19  R20  R22  R24
 R18:  R25
 R19:  R17  R21  R23  R25
 R20:  R6   R17  R21  R23  R25
 R21:  R5   R11  R14  R16  R19  R20  R22  R24
 R22:  R4   R7   R10  R17  R21  R23  R25
 R23:  R5   R11  R14  R15  R16  R19  R20  R22  R24
 R24:  R7   R8   R10  R12  R13  R17  R21  R23  R25
 R25:  R2   R3   R5   R11  R14  R16  R18  R19  R20  R22
       R24

 Current process size: 234424 kB
```

Afterwards, additional additive properties can be read from file. A property file contains the number of properties, the name and type of the properties (1 real, 0 integer) and the data, which can be mapped using the fragment name:

```
@nof_properties 2
@property price 1
@property importance 0
@values
CORE1    11    22
CORE2    13.02 24
```

We can read the properties using the `ADDPROP` command.

```
LOFT/FRAGSPACE> ADDPROP  /home/goofy/prop
   2 properties read in:
     price (double, 2 values)
     importance (int, 2 values)
```

To restrict the search space, a filter can be defined.

```
  Molecules can be filtered using:
   Properties:
   MW[min, max]:                    MW
   TPSA[min, max]:                  TPSA
   Atoms[min, max]:                 # Non-hydrogen atoms
   Acceptors[min, max]:             # Acceptors
   Donors[min, max]:                # Donors
   Hetero[min, max]:                # Hetero atoms
   AromAtoms[min, max]:             # Aromatic atoms
   Halogens[min, max]:              # Halogens
   Inorganic[min, max]:             # Inorganic atoms
   Bonds[min, max]:                 # Non-hydrogen bonds
   RotB[min, max]:                  # Rot. bonds
   CRTB[min, max]:                  Max CRTB
   Ringsystems[min, max]:           # Ringsystems
   AroRingsystems[min, max]:        # Arom. ringsystems
   Rings[min, max]:                 # Rings
   AroRings[min, max]:              # Arom. rings
   MaxRSsize[min, max]:             Max atoms in ringsys
   RS[min, max]:                    # R/S centers
   EZ[min, max]:                    # E/Z bonds
   Linkers[min, max]:               # Link atoms
   Charge[min, max]:                Charge
   PLogP[min, max]:                 PLogP
   Volume[min, max]:                Volume
   Smarts['expression'][min, max]:  Smarts
   Func['expression'][min, max]:    Functional group
   LinkName['expression'][min, max]: Occ. of a link name
  Operators:
   AND: A and B -> both expressions must be true
   OR:  A or B  -> A or B must be true
   NOT: not A   -> A must be false
   TOLERANCE[x]{A, B, C}: x of  A, B or C are allowed to be false
  Examples:
   MW[200, 400] or not Charge[-10,-1]: Either Molecular weight in between 200 and 400 or charge is positive
   TOLERANCE[1]{Donors[0,5], Acceptors[0,10], MW[0,500], PLogP[-20,5]}: the Lipinski 'rule of 5'
```

In the following case, we define a filter for all reagents (case 0):

```
LOFT/FRAGSPACE> FILTER
LOFT/FILTER>ENTER  0 "MW[0, 500] or TPSA[0,499]"
  Old reagents filter: None
  New reagents filter:  ( (MW[0,500] OR TPSA[0,499]))
```

To focus the libraries on a query, we change to the query submenu and load some compounds from file.

```
LOFT/QUERY> READ  /home/goofy/data/TEST 10 11
   Added 2 query molecules from file /home/goofy/data/TEST.mol2
   2 query molecules loaded altogether.
LOFT/QUERY> qsel

    Slot (anti)query
    ----------------------------
     1 ten
     2 eleven
    Selected queries:
     None.
    Selected antiqueries:
     None.
    Select queries (e.g. 1-2, 7, 13; 14) <1-2> [1] : 1-2
    Set equal weights? [y] :
```

After we have chosen two queries, we cluster the reagent to achieve higher diversity within the designed libraries. *LOFT* provides a wizard, guiding you through the clustering process. Note, that the computation time increases with each chosen query. Normally, a single query optimization is performed.

```
LOFT/QUERY> CLUSTER
LOFT/CLUSTER> WIZARD
   Mode:
    0 use all reagents
    1 reagents of certain link types
    2 reagents compatible to one of these link types
   Reagents will be selected using filter  ( (MW[0,500] OR TPSA[0,499]))
   Clustering data initialized for 85 reagents.
   Computing distances between reagents
   (     85/   3570)
   Verifying distance matrix...
   Compute complete linkage clustering with a maximum distance of    0
   45 clustering steps performed.
    Number of clusters:          23
    Average size:                 2.957
    Std. Dev.:                    1.301
    Number in clusters:          68(80.000%)
    Number of singletons:        17(20.000%)
   Printing the cluster ids of the recently performed clustering:
   0x5e32ac0b 1
   0xbb87fe6b 1
   0x1ffdabc7 2
   .
   .
   .
    Process time used:  0.14 s.    Current process size: 234820 kB
```

Now, we change to the main menu of *LOFT*, where the optimization routines can be started.

```
LOFT> FOCUS
```

The scoring function can be defined using the command `SCORING` (see also figure 1.3). Here, we set feature tree similarity as the only criterium, allowing similarity from 0.0 to 1.0.

```
LOFT/FOCUS> SCORING ftsim 1.0 0 1
   #    Property     |     mode       |  weight  | left frontier |  left  |  right  |  right frontier
   ---------------------------------------------------------------------------------------------------
    1 Charge          score uniformly    0.0000          0            0         0            0
    2 importance      score uniformly    0.0000          0            0         0            0
    3 Atoms           score uniformly    0.0000          0            0         0            0
    4 Acceptors       score uniformly    0.0000          0            0         5            5
    5 Donors          score uniformly    0.0000          0            0         8            8
    6 Hetero          score uniformly    0.0000          0            0         0            0
    7 AromAtoms       score uniformly    0.0000          0            0         0            0
```

```
    8 Halogens        score uniformly    0.0000          0            0         0            0
    9 Inorganic       score uniformly    0.0000          0            0         0            0
   10 Bonds           score uniformly    0.0000          0            0         0            0
   11 RotB            score uniformly    0.0000          0            0         0            0
   12 CRTB            score uniformly    0.0000          0            0         0            0
   13 Ringsystems     score uniformly    0.0000          0            0         0            0
   14 AroRingsystems  score uniformly    0.0000          0            0         0            0
   15 Rings           score uniformly    0.0000          0            0         4            4
   16 AroRings        score uniformly    0.0000          0            0         0            0
   17 MaxRSsize       score uniformly    0.0000          0            0         0            0
   18 RS              score uniformly    0.0000          0            0         0            0
   19 EZ              score uniformly    0.0000          0            0         0            0
   20 Linkers         score uniformly    0.0000          0            0         0            0
   21 MW              score uniformly    0.0000          0.0000     200.0000  450.0000     500.0000
   22 TPSA            score uniformly    0.0000          0.0000       0.0000    0.0000       0.0000
   23 PLogP           score uniformly    0.0000          0.0000       0.0000    0.0000       0.0000
   24 Volume          score uniformly    0.0000          0.0000       0.0000    0.0000       0.0000
   25 price           score uniformly    0.0000          0.0000       0.0000    0.0000       0.0000
   26 FtSim           add score          1.0000          0.0000 <= Sim to query          <=  1.0000
   27 FtDis           add (1 - score)    0.0000          0.0000 <= Dissim to antiquery   <=  1.0000
```

Now we pick some cores with the same link patterns from the fragment space, doing a multi-core optimization. The cores can be weighted. Per default the maximum similarity to any of the cores is used. Again, the computation time increases with each chosen core. Normally a single core optimization is performed.

```
LOFT/FOCUS> PICK 86 y "82 84"
   Selected cores are (Select other IDs to overwrite or an invalid one to drop it, anything else to abort):
      none.

   Possible cores are (according to the core filter):
       Id | Name                |  # | Link types
        1 | CORE1               |  2 | R11 R10
       62 | CORE2               |  2 | R5 R4
       63 | CORE3               |  3 | R16 R15 R14
       81 | CORE4               |  2 | R18 R17
       82 | CORE5               |  2 | R18 R17
       83 | CORE6               |  2 | R18 R17
       84 | CORE7               |  2 | R18 R17
       85 | CORE8               |  2 | R18 R17
       86 | CORE9               |  2 | R18 R17       .
        .
        .
   Core CORE9 selected.
   Possible cores with the same link patterns are:
       Id | Name                |  # | Link types
        1 | CORE1               |  2 | R11 R10
       62 | CORE2               |  2 | R5 R4
       63 | CORE3               |  3 | R16 R15 R14
       81 | CORE4               |  2 | R18 R17
       82 | CORE5               |  2 | R18 R17
       83 | CORE6               |  2 | R18 R17
       84 | CORE7               |  2 | R18 R17
       85 | CORE8               |  2 | R18 R17
       86 | CORE9               |  2 | R18 R17
   Selected cores:
    CORE9
    CORE7
    CORE5
```

As diversity measurement, we take the cluster ids (1). If more than two reagents from one cluster occur (2) we penalize with 0.1.

```
LOFT/FOCUS> DIVERSE  1 0.1 1 1 2
```

```
   Diversity measurements for reagents of the same core link:
   Note, that a distance value of 0.1 means a maximum similarity of 0.9
   ------------------------------------------------------------------------

      Penalty function:
         B----C       max penalty(<= 1.0)
        /      \        % penalty
     __A        D__   0.0

    01 cluster ids: disabled.
    02 average distance: disabled.
    03 pairwise distance: disabled.
```

In the next step we add the number of rotatable bonds to the scoring function and decrease the weight of the ftree similarity so the weights sum up to 1.0.

```
LOFT/FOCUS> SCORING  ROTB 0.2 1 0 100 100 100
   #    Property     |     mode       |  weight  | left frontier |  left  |  right  |  right frontier
   ---------------------------------------------------------------------------------------------------
    1 Charge          score uniformly    0.0000          0            0         0            0
    2 importance      score uniformly    0.0000          0            0         0            0
    3 Atoms           score uniformly    0.0000          0            0         0            0
    4 Acceptors       score uniformly    0.0000          0            0         5            5
    5 Donors          score uniformly    0.0000          0            0         8            8
    6 Hetero          score uniformly    0.0000          0            0         0            0
    7 AromAtoms       score uniformly    0.0000          0            0         0            0
    8 Halogens        score uniformly    0.0000          0            0         0            0
    9 Inorganic       score uniformly    0.0000          0            0         0            0
   10 Bonds           score uniformly    0.0000          0            0         0            0
   11 RotB            score uniformly    0.0000          0            0         0            0
   12 CRTB            score uniformly    0.0000          0            0         0            0
   13 Ringsystems     score uniformly    0.0000          0            0         0            0
   14 AroRingsystems  score uniformly    0.0000          0            0         0            0
   15 Rings           score uniformly    0.0000          0            0         4            4
   16 AroRings        score uniformly    0.0000          0            0         0            0
   17 MaxRSsize       score uniformly    0.0000          0            0         0            0
   18 RS              score uniformly    0.0000          0            0         0            0
   19 EZ              score uniformly    0.0000          0            0         0            0
   20 Linkers         score uniformly    0.0000          0            0         0            0
   21 MW              score uniformly    0.0000          0.0000     200.0000  450.0000     500.0000
   22 TPSA            score uniformly    0.0000          0.0000       0.0000    0.0000       0.0000
   23 PLogP           score uniformly    0.0000          0.0000       0.0000    0.0000       0.0000
   24 Volume          score uniformly    0.0000          0.0000       0.0000    0.0000       0.0000
   25 price           score uniformly    0.0000          0.0000       0.0000    0.0000       0.0000
   26 FtSim           add score          1.0000          0.0000 <= Sim to query          <=  1.0000
   27 FtDis           add (1 - score)    0.0000          0.0000 <= Dissim to antiquery   <=  1.0000

   Sum of weights:1.0000
   Desirability function:
       B----C       1.0
      /      \         score
   __A        D__   0.0

   New sum of weights: 1.2000
FOCUS> SCORING  Ftsim 0.8
   #    Property     |     mode       |  weight  | left frontier |  left  |  right  |  right frontier
   ---------------------------------------------------------------------------------------------------
    1 Charge          score uniformly    0.0000          0            0         0            0
    .
   11 RotB            score uniformly    0.2000          0          100       100          100
    .
.   .
   26 FtSim           add score          1.0000          0.0000 <= Sim to query        <=   1.0000
   27 FtDis           add (1 - score)    0.0000          0.0000 <= Dissim to antiquery <=   1.0000

  New sum of weights: 1.0000
```

Eventually we are ready for computation. In this tutorial, we start now a simulated annealing.

```
LOFT/FOCUS> ANNEALING
   Selected queries:
    ten    (0.5000)
    eleven (0.5000)
   Select fragments:
   > Reagent 0x5bbff4c6 is incompatible
   > Reagent 0x7c67f6d7 is incompatible
   .
   .
   .


      (   101/   101)     25 reagents rejected
   Settings:
    Output dir:                        /home/fischer/tmp/
    Scoring:                           arithmetic
    Steps to go:                       1000
    Nof max unsuccessful steps:        250
    Nof solutions per core:            3
    Seed mode:                         take user defined seed
    Seed:                              1
    FTree reagent size filter:         0.7500 1.5000
     (min*nof_nodes(query) <= x <= max* nof_nodes(query))
    Product size filter:               0.2000 1.8000
     (min*nof_heavy_atoms(query) <= x <= max* nof_heavy_atoms(query))
    FTrees regioselectivity factor:    0.0000
    Select start reagents by:          random choice
    Max nof reagents for each
        core link to consider:         1000000
    Algorithm:                         simulated annealing
    Temperature:                       1.00000000
    Expon. cooling:                    0.99950000
   Number of core sets: 1
   Number of reagents: 60

   R1    >      0
   .
   .
   .
   R18   >      0
   R19   >      8
   R20   >      2
   R21   >      0
   R22   >     10
   R23   >      0
   R24   >     10
   R25   >     30

   ---> Core(s):
   0x73f27bd3 (max)
   0x275db280 (max)
   0x30384edb (max)
   Number of core links: 2

   Select random start reagents for each core link


   ------------------------------------------------------

          Start simulated annealing
   ------------------------------------------------------


   Number of possible solutions:
   Core link R18: 10 out of 30 reagents (0 preselected)
   Core link R17: 10 out of 30 reagents (0 preselected)
```

```
   Number of possible solutions altogether:
   9e+14

   Generating temporary data.
   > Generate start state
   > R18:
   >  0xebb7ef67           (R25    11)
   .
   .
   .
   > R17:
   >  0x8784277a           (R22    13)
   .
   .
   .

   Score for start solution: 0.5898
   > R18  : Substitute      0x79b81d4a       (R25      7) with score 0.5958
                        for 0xe2ae6c02       (R25    -  ) with score 0.5898
   >   R17: Substitute      0x688efef0       (R22     13) with score 0.5965
                        for 0xcdc4205e       (R22     13) with score 0.5958
   >   R18: Substitute      0xf5b6fc54       (R25      6) with score 0.5997
                        for 0xc87ff4e9       (R25     11) with score 0.5965
   >   R18: Substitute      0xf3572bcf       (R25      3) with score 0.5997
                        for 0x9002b0e6       (R25      3) with score 0.5997
   >   R17: Substitute      0x8b8ce0a4       (R24     21) with score 0.6152
                        for 0x4be0942        (R22     22) with score 0.5997
   >   R17: Substitute      0xecc705b0       (R24    -  ) with score 0.6153
                        for 0xe92784f8       (R24      9) with score 0.6152
   >   R18: Substitute      0xcf44348c       (R25    -  ) with score 0.6156

    (    1000/    1000)   Best score: 0.6217
   Steps used: 1000

 ------------------------------------------------
 Solution no 1 with score 0.6217, seen after 1000 steps
  Cores used:
    CORE9
    CORE7
    CORE5
  .
  .
  .
```

Finally we write the results and exit.

```
LOFT/FOCUS> write
   Selectable core sets:
   1 CORE9(max) CORE7(max) CORE5(max)
   Core(s):
    CORE9
    CORE7
    CORE5
   ------------------------------------------------------------
   Write solution no 1 with score 0.6217, seen after 475 steps:
   Enumerate products...
   Writing file /home/goofy/tmp/eleven-CORE9_ANNEAL_01_01_110113-211041.pdf
   .
   .
   .
LOFT/FOCUS> QUIT
 >> Releasing user data.
 >> Releasing system data.
 >> Bye!
```

The whole example we went through is shown in the following script:

```
#Example batchfile for LOFT 0.97

# redefining default values (comment)
SET NOF_SOL 3 # remember the best 3 solutions seen

# read the fragspace and additional properties
FRAGSPACE
 READ /home/goofy/data/TestSpace.fsf
 ADDPROP /home/goofy/prop

# filter the reagents )
FILTER
 ENTER  0 "MW[0, 500] or TPSA[0,499]"

# read a query
QUERY
 read /home/goofy/data/TEST.mol2 10 11
 QSEL 1-2

# create cluster ids for reagents
CLUSTER
 WIZARD


# generate focussed library with simulated annealing
FOCUS
 PICK 86 y "82 84"
 DIVERSE 1 0.1 1 2 # use cluster id diversity measurement, penalizing reagents with 0.1
 set MAX_STEPS 1000
 SCORING ROTB 0.2 1  0 100 100 100
 SCORING Ftsim 0.8
 ANNEALING
 WRITE
QUIT
```

We can execute this workflow also by executing *LoFT* with the '-b' option:

```
 LoFT -b example.bat
```

Also, a script can be called using the 'SCRIPT' command

```
LOFT> SCRIPT example.bat
```

Bibliography

[1] I. Dramburg, S. Hindle, and M. Lilienthal. *CoLibri 1.3 User Guide*. BioSolveIT GmbH, St. Augustin, Germany, 2011. 156

[2] J.R. Fischer, U. Lessel, and M. Rarey. Loft: Similarity-driven multiobjective focused library design. *J. Chem. Inf. Model.*, 50(1):1–21, 2010. 155, 159

[3] M. Rarey and J.S. Dixon. Feature trees: a new molecular similarity measure based on tree matching. *Journal of Computer-Aided Molecular Design*, 12(5):471–490, 1998. 159

[4] M. Rarey and M. Stahl. Similarity searching in large combinatorial chemistry spaces. *J. Comput.-Aided Mol. Des.*, 15(6):497–520, 2001. 159

[5] M. Rarey, M. Zimmermann, S. Hindle, M. Gastreich, and R. Fischer. *FTrees 2.3 User Guide*. BioSolveIT GmbH, St. Augustin, Germany, 2011. 159

[6] TIBCO Spotfire, Somerville, MA, USA. *TIBCO Spotfire DecisionSite 9.1.1*, 2008. 157

Literaturverzeichnis

[1] RCSB Protein Data Bank. **PDB** [online]. http://www.pdb.org/pdb/home/home.do [Abruf: 02.01.2011]. - Zitiert auf den Seiten 1 und 31.

[2] S. Rowsell, P. Hawtin, C.A. Minshull, H. Jepson, S.M.V. Brockbank, D.G. Barratt, A.M. Slater, W.L. McPheat, D. Waterson, A.M. Henney und R.A. Pauptit. **Crystal structure of human MMP9 in complex with a reverse hydroxamate inhibitor**. *Journal of Molecular Biology*, **319**(1):173–181, 2002. doi:10.1016/S0022-2836(02)00262-0. - Zitiert auf Seite 1.

[3] K. Stierand, P.C. Maass und M. Rarey. **Molecular complexes at a glance: automated generation of two-dimensional complex diagrams**. *Bioinformatics (Oxford, England)*, **22**(14):1710–1716, 2006. doi:10.1093/bioinformatics/btl150. - Zitiert auf Seite 2.

[4] L.B. Kier. **Molecular orbital calculation of preferred conformations of acetylcholine, muscarine, and muscarone**. *Molecular Pharmacology*, **3**(5):487–494, 1967. - Zitiert auf Seite 1.

[5] L.B. Kier. *MO Theory in Drug Research*. Academic Press, New York, 1971. - Zitiert auf Seite 1.

[6] J.H. Van Drie. **Monty Kier and the Origin of the Pharmacophore Concept**. *Internet Electronic Journal of Molecular Design*, **6**:271–279, 2007. - Zitiert auf Seite 1.

[7] H.-J. Böhm, G. Klebe und H. Kubinyi. *Wirkstoffdesign*. Spektrum Akademischer Verlag, 2002. - Zitiert auf den Seiten 2 und 95.

[8] E. Fischer. **Einfluss der Configuration auf die Wirkung der Enzyme**. *Berichte der deutschen chemischen Gesellschaft*, **27**(3):2985–2993, 1894. doi:10.1002/cber.18940270364. - Zitiert auf Seite 2.

[9] D.K. Agrafiotis, V.S. Lobanov und F.R. Salemme. **Combinatorial informatics in the post-genomics Era**. *Nature Reviews. Drug Discovery*, **1**(5):337–346, 2002. - Zitiert auf den Seiten 3 und 7.

[10] V.J. Gillet. **Applications of Evolutionary Computation in Drug Design**. *Structure & Bonding*, **110**:133–152, 2004. doi:10.1007/b13935. - Zitiert auf den Seiten 2, 5, 14 und 22.

[11] E.G. Johnson und G.M. Maggiora. *Concepts and Applications of Molecular Similarity*. Wiley, New York, 1990. - Zitiert auf den Seiten 2 und 20.

[12] D.K. Agrafiotis und V.S. Lobanov. **Ultrafast algorithm for designing focused combinational arrays**. *Journal of Chemical Information and Computer Sciences*, **40**(4):1030–1038, 2000. - Zitiert auf den Seiten 2 und 23.

[13] R.E. Dolle. **Historical overview of chemical library design**. *Methods in Molecular Biology (Clifton, N.J.)*, **685**:3–25, 2011. doi:10.1007/978-1-60761-931-4_1. - Zitiert auf Seite 2.

[14] H. van de Waterbeemd und E. Gifford. **ADMET in silico modelling: towards prediction paradise?** *Nature Reviews. Drug Discovery*, **2**(3):192–204, 2003. doi:10.1038/nrd1032. - Zitiert auf Seite 3.

[15] J.G. Lombardino und J.A. 3rd Lowe. **The role of the medicinal chemist in drug discovery–then and now**. *Nature Reviews. Drug Discovery*, **3**(10):853–862, 2004. doi:10.1038/nrd1523. - Zitiert auf Seite 3.

[16] K. Frobel und T. Kraemer. **Kombinatorische Synthese**. *Chemie in unserer Zeit*, **30**(6):270–285, 1996. doi:10.1002/ciuz.19960300603. - Zitiert auf den Seiten 4, 8, 9 und 50.

[17] H.J. Boehm und G. Schneider. *Virtual Screening for Bioactive Molecules*. John Wiley & Sons, Inc., 2000. - Zitiert auf Seite 4.

[18] F. Schueth. **Hochdurchsatz-Untersuchungen**. In R. Dittmeyer, W. Keim, G. Kreysa und A. Oberholz, Hrsg., *Winnacker-Küchler: Chemische Technik: Prozesse und Produkte*, **2**. Wiley-VCH, 2004. - Zitiert auf den Seiten 4, 7, 8, 9 und 10.

[19] E.J. Martin, J.M. Blaney, M.A. Siani, D.C. Spellmeyer, A.K. Wong und W.H. Moos. **Measuring diversity: experimental design of combinatorial libraries for drug discovery**. *Journal of Medicinal Chemistry*, **38**(9):1431–1436, 1995. - Zitiert auf den Seiten 4 und 20.

[20] D.K. Agrafiotis. *Diversity of chemical libraries*. John Wiley and Sons, Chichester, 1998. - Zitiert auf den Seiten 4, 14, 20 und 21.

[21] D.K. Agrafiotis, J.C. Myslik und F.R. Salemme. **Advances in diversity profiling and combinatorial series design**. *Molecular Diversity*, **4**(1):1–22, 1998. - Zitiert auf den Seiten 4 und 20.

[22] H. Matter und M. Rarey. **Design and Diversity Analysis of Compound Libraries for Lead Discovery**. In G. Jung, Hrsg., *Combinatorial Organic Chemistry*, S. 409–439. Wiley-VCH, 1999. - Zitiert auf den Seiten 4, 20 und 29.

[23] L. Weber. **Current Status of Virtual Combinatorial Library Design**. *QSAR & Combinatorial Science*, **24**(7):809–823, 2005. doi:10.1002/qsar.200510120. - Zitiert auf den Seiten 4, 7, 14, 20, 21 und 58.

[24] D.M. Schnur. **Recent trends in library design: 'rational design' revisited**. *Current Opinion in Drug Discovery & Development*, **11**(3):375–380, 2008. - Zitiert auf den Seiten 4 und 22.

[25] D.M. Schnur, B.R. Beno, A.J. Tebben und C. Cavallaro. **Methods for combinatorial and parallel library design**. *Methods in Molecular Biology (Clifton, N.J.)*, **672**:387–434, 2011. doi:10.1007/978-1-60761-839-3_16. - Zitiert auf den Seiten 4, 7 und 22.

LITERATURVERZEICHNIS

[26] V.J. Gillet, W. Khatib, P. Willett, P.J. Fleming und D.V.S. Green. **Combinatorial library design using a multiobjective genetic algorithm**. *Journal of Chemical Information and Computer Sciences*, **42**(2):375–385, 2002. - Zitiert auf den Seiten 4, 15 und 23.

[27] E.A. Jamois, M. Hassan und M. Waldman. **Evaluation of reagent-based and product-based strategies in the design of combinatorial library subsets**. *Journal of Chemical Information and Computer Sciences*, **40**(1):63–70, 2000. - Zitiert auf den Seiten 5 und 14.

[28] E.A. Jamois. **Reagent-based and product-based computational approaches in library design**. *Current Opinion in Chemical Biology*, **7**(3):326–330, 2003. - Zitiert auf den Seiten 5 und 14.

[29] P. Willett. **Chemoinformatics – similarity and diversity in chemical libraries**. *Current Opinion in Biotechnology*, **11**(1):85–88, 2000. - Zitiert auf den Seiten 5, 14 und 20.

[30] M. Rarey und J.S. Dixon. **Feature trees: a new molecular similarity measure based on tree matching**. *Journal of Computer-Aided Molecular Design*, **12**(5):471–490, 1998. - Zitiert auf den Seiten 6, 17, 29, 30, 31, 34, 35, 37, 39, 46 und 143.

[31] M. Rarey und M. Stahl. **Similarity searching in large combinatorial chemistry spaces**. *Journal of Computer-Aided Molecular Design*, **15**(6):497–520, 2001. - Zitiert auf den Seiten 6, 24, 25, 26, 27, 34, 36, 54, 64, 73 und 76.

[32] J.R. Fischer, U. Lessel und M. Rarey. **LoFT: Similarity-Driven Multiobjective Focused Library Design**. *Journal of Chemical Information and Modeling*, **50**(1):1–21, 2010. doi:10.1021/ci900287p. - Zitiert auf den Seiten 6, 68, 79, 89, 113 und 143.

[33] J.R. Fischer, U. Lessel und M. Rarey. **Improving similarity-driven library design: customized matching and regioselective feature trees**. *Journal of Chemical Information and Modeling*, **51**(9):2156–2163, 2011. doi:10.1021/ci200014g. - Zitiert auf den Seiten 6 und 69.

[34] J. Degen und M. Rarey. **FlexNovo: structure-based searching in large fragment spaces**. *ChemMedChem*, **1**(8):854–868, 2006. doi:10.1002/cmdc.200500102. - Zitiert auf den Seiten 6, 26, 76, 80, 123 und 143.

[35] S. Urbaczek, A. Kolodzik, R. Fischer, T. Lippert und M. Rarey. **NAOMI - On the almost trivial task of reading molecules from different file formats**. *Journal of Chemical Information and Modeling*, 2011. - Zitiert auf den Seiten 6, 41, 64, 81 und 144.

[36] G. Schneider. **Trends in virtual combinatorial library design**. *Current Medicinal Chemistry*, **9**(23):2095–2101, 2002. - Zitiert auf Seite 7.

[37] *Combinatorial Library Design and Evaluation: Principles, Software, Tools, and Applications in Drug Discovery*. CRC Press, 2001. - Zitiert auf Seite 7.

[38] G. Jung, Hrsg. *Combinatorial Organic Chemistry*. Wiley-VCH, 1999. - Zitiert auf Seite 7.

[39] V.J. Gillet. *Computational Medicinal Chemistry for Drug Discovery*, Kapitel Computational Aspects of Library Design. Marcel Dekker, 2004. - Zitiert auf Seite 7.

[40] V.J. Gillet. **New directions in library design and analysis**. *Current Opinion in Chemical Biology*, **12**(3):372–378, 2008. doi:10.1016/j.cbpa.2008.02.015. - Zitiert auf den Seiten 7 und 22.

[41] Joe Zhongxiang Zhongxiang Zhou, Hrsg. *Chemical Library Design (Methods in Molecular Biology)*. Humana Press, 2010. - Zitiert auf Seite 7.

[42] G. Klebe. *Wirkstoffdesign: Entwurf und Wirkung von Arzneistoffen*. Spektrum Akademischer Verlag, 2009. - Zitiert auf Seite 7.

[43] W.A. Warr. **Some Trends in Chem(o)informatics**. *Methods in Molecular Biology (Clifton, N.J.)*, **672**:1–37, 2011. doi:10.1007/978-1-60761-839-3_1. - Zitiert auf Seite 7.

[44] S. Braese und B. Neuss. **Glossar von Begriffen der Kombinatorischen Chemie**. *Angewandte Chemie*, **114**(5):893–906, 2002. doi:10.1002/1521-3757(20020301)114:5<893::AID-ANGE893>3.0.CO;2-S. - Zitiert auf den Seiten 7 und 11.

[45] F. Balkenhohl und von dem Bussche-H. **Kombinatorische Synthese niedermolekularer organischer Verbindungen**. *Angewandte Chemie*, **108**(20):2436–2488, 1996. doi:10.1002/ange.19961082004. - Zitiert auf Seite 9.

[46] B.K. Shoichet. **Virtual screening of chemical libraries**. *Nature*, **432**(7019):862–865, 2004. doi:10.1038/nature03197. - Zitiert auf Seite 10.

[47] G. Klebe. **Virtual ligand screening: strategies, perspectives and limitations**. *Drug Discovery Today*, **11**(13-14):580–594, 2006. doi:10.1016/j.drudis.2006.05.012. - Zitiert auf Seite 10.

[48] T. Lengauer, C. Lemmen, M. Rarey und M. Zimmermann. **Novel technologies for virtual screening**. *Drug Discovery Today*, **9**(1):27–34, 2004. - Zitiert auf den Seiten 10 und 29.

[49] I. Muegge und M. Rarey. *Reviews in Computational Chemistry*, **17**, Kapitel Small Molecule Docking and Scoring, S. 1–60. Wiley-VCH, 2001. - Zitiert auf Seite 10.

[50] N.S. Gray, L. Wodicka, A.M. Thunnissen, T.C. Norman, S. Kwon, F.H. Espinoza, D.O. Morgan, G. Barnes, S. LeClerc, L. Meijer, S.H. Kim, D.J. Lockhart und P.G. Schultz. **Exploiting chemical libraries, structure, and genomics in the search for kinase inhibitors**. *Science (New York, N.Y.)*, **281**(5376):533–538, 1998. - Zitiert auf den Seiten 11 und 96.

[51] B.A. Leland, B.D. Christie, J.G. Nourse, D.L. Grier, R.E. Carhart, T. Maffett, S.M. Welford und D.H. Smith. **Managing the Combinatorial Explosion**. *Journal of Chemical Information and Modeling*, **37**(1):62–70, 1997. doi:10.1021/ci960088t. - Zitiert auf Seite 10.

[52] J.Z. Zhou. **Chemoinformatics and library design**. *Methods in Molecular Biology (Clifton, N.J.)*, **685**:27–52, 2011. doi:10.1007/978-1-60761-931-4_2. - Zitiert auf Seite 11.

[53] B.R. Beno und J.S. Mason. **The design of combinatorial libraries using properties and 3D pharmacophore fingerprints**. *Drug Discovery Today*, **6**(5):251–258, 2001. - Zitiert auf den Seiten 11 und 14.

[54] A.G. Maldonado, J.P. Doucet, M. Petitjean und B.-T. Fan. **Molecular similarity and diversity in chemoinformatics: from theory to applications**. *Molecular Diversity*, **10**(1):39–79, 2006. doi:10.1007/s11030-006-8697-1. - Zitiert auf Seite 11.

[55] E.J. Martin und R.E. Critchlow. **Beyond mere diversity: tailoring combinatorial libraries for drug discovery**. *Journal of Combinatorial Chemistry*, **1**(1):32–45, 1999. - Zitiert auf Seite 12.

[56] D. Tiebes. **Combinatorial Chemistry**. In G. Jung, Hrsg., *Combinatorial Chemistry: Synthesis, Analysis, Screening*, Kapitel 1. Wiley-VCH, 1999. doi:10.1002/9783527613502. - Zitiert auf Seite 12.

[57] D.K. Agrafiotis. **Multiobjective optimization of combinatorial libraries**. *Journal of Computer-Aided Molecular Design*, **16**(5-6):335–356, 2002. - Zitiert auf den Seiten 14 und 89.

[58] V.J. Gillet, P. Willett und J. Bradshaw. **The Effectiveness of Reactant Pools for Generating Structurally-Diverse Combinatorial Libraries**. *Journal of Chemical Information and Computer Sciences*, **37**(4):731–740, 1997. - Zitiert auf Seite 14.

[59] V.J. Gillet und O. Nicolotti. **Evaluation of reactant-based and product-based approaches to the design of combinatorial libraries**. *Perspectives in Drug Discovery and Design*, **20**:265–287, 2000. - Zitiert auf Seite 14.

[60] S. Kirkpatrick, C.D. Gelatt und M.P. Vecchi. **Optimization by Simulated Annealing**. *Science (New York, N.Y.)*, **220**(4598):671–680, 1983. doi:10.1126/science.220.4598.671. - Zitiert auf den Seiten 14 und 53.

[61] R.P. Sheridan, S.G. SanFeliciano und S.K. Kearsley. **Designing targeted libraries with genetic algorithms**. *Journal of Molecular Graphics & Modelling*, **18**(4-5):320–34, 525, 2000. - Zitiert auf Seite 15.

[62] C.A. Nicolaou und C.C. Kannas. **Molecular library design using multi-objective optimization methods**. *Methods in Molecular Biology (Clifton, N.J.)*, **685**:53–69, 2011. doi:10.1007/978-1-60761-931-4_3. - Zitiert auf den Seiten 15 und 24.

[63] C.A. Nicolaou, N. Brown und C.S. Pattichis. **Molecular optimization using computational multiobjective methods**. *Current Opinion in Drug Discovery & Development*, **10**(3):316–324, 2007. - Zitiert auf Seite 15.

[64] J. Harrington. **The Desirability Function**. *Industrial Quality Control*, **21**(10):494, 1965. - Zitiert auf den Seiten 16, 23 und 55.

[65] G. Derringer und R. Suich. **Simultaneous-Optimization Of Several Response Variables**. *Journal Of Quality Technology*, **12**(4):214–219, 1980. - Zitiert auf den Seiten 16, 23, 55 und 146.

[66] H. Trautmann und C. Weihs. **On the distribution of the desirability index using Harrington's desirability function**. *Metrika*, **63**(2):207–213, 2006. doi:10.1007/s00184-005-0012-0. - Zitiert auf den Seiten 16 und 55.

[67] R. Mannhold, H. Kubinyi und G.Folkers, Hrsg. *Molecular Drug Properties: Measurement and Prediction (Methods and Principles in Medicinal Chemistry)*. Wiley-VCH, 2007. - Zitiert auf den Seiten 16 und 18.

[68] J. Paern, J. Degen und M. Rarey. **Exploring fragment spaces under multiple physicochemical constraints**. *Journal of Computer-Aided Molecular Design*, **21**(6):327–340, 2007. doi:10.1007/s10822-007-9121-3. - Zitiert auf den Seiten 16, 26, 61 und 143.

[69] J.-F. Truchon und C.I. Bayly. **GLARE: a new approach for filtering large reagent lists in combinatorial library design using product properties**. *Journal of Chemical Information and Modeling*, **46**(4):1536–1548, 2006. doi:10.1021/ci0504871. - Zitiert auf den Seiten 16 und 23.

[70] J.-F. Truchon. **GLARE: A tool for product-oriented design of combinatorial libraries**. *Methods in Molecular Biology (Clifton, N.J.)*, **685**:337–346, 2011. doi:10.1007/978-1-60761-931-4_17. - Zitiert auf den Seiten 16 und 23.

[71] MDL. **MACCS II Manual** [online]. http://www.mdli.com/ [Abruf: 07.01.2011]. - Zitiert auf Seite 16.

[72] Daylight. **Daylight Theory Manual** [online]. http://www.daylight.com/ [Abruf: 01.01.2011]. - Zitiert auf den Seiten 16, 41 und 64.

[73] L. Xing und R.C. Glen. **Novel methods for the prediction of logP, pK(a), and logD**. *Journal of Chemical Information and Computer Sciences*, **42**(4):796–805, 2002. - Zitiert auf den Seiten 17 und 123.

[74] A. Bender, H.Y. Mussa, R.C. Glen und S. Reiling. **Molecular similarity searching using atom environments, information-based feature selection, and a naive Bayesian classifier**. *Journal of Chemical Information and Computer Sciences*, **44**(1):170–178, 2004. doi:10.1021/ci034207y. - Zitiert auf den Seiten 17 und 123.

[75] D. Rogers und M. Hahn. **Extended-connectivity fingerprints**. *Journal of Chemical Information and Modeling*, **50**(5):742–754, 2010. doi:10.1021/ci100050t. - Zitiert auf den Seiten 17 und 123.

[76] Accelrys [online]. http://accelrys.com/ [Abruf: 07.01.2011]. - Zitiert auf den Seiten 17 und 123.

[77] P. Willett, J.M. Barnard und G.M. Downs. **Chemical Similarity Searching**. *Journal of Chemical Information and Computer Sciences*, **38**(6):983–996, 1998. doi:10.1021/ci9800211. - Zitiert auf Seite 17.

[78] X. Chen und C.H. Reynolds. **Performance of similarity measures in 2D fragment-based similarity searching: comparison of structural descriptors and similarity coefficients**. *Journal of Chemical Information and Computer Sciences*, **42**(6):1407–1414, 2002. - Zitiert auf Seite 17.

[79] R.D. Brown und Y.C. Martin. **Use of Structure-Activity Data To Compare Structure-Based Clustering Methods and Descriptors for Use in Compound Selection**. *Journal of Chemical Information and Modeling*, **36**(3):572–584, 1996. doi:10.1021/ci9501047. - Zitiert auf Seite 17.

[80] R.D. Brown und Y.C. Martin. **The Information Content of 2D and 3D Structural Descriptors Relevant to Ligand-Receptor Binding**. *Journal of Chemical Information and Modeling*, **37**(1):1–9, 1997. doi:10.1021/ci960373c. - Zitiert auf Seite 17.

[81] H. Matter. **Selecting optimally diverse compounds from structure databases: a validation study of two-dimensional and three-dimensional molecular descriptors**. *Journal of Medicinal Chemistry*, **40**(8):1219–1229, 1997. doi:10.1021/jm960352+. - Zitiert auf Seite 17.

[82] Y.C. Martin, J.L. Kofron und L.M. Traphagen. **Do structurally similar molecules have similar biological activity?** *Journal of Medicinal Chemistry*, **45**(19):4350–4358, 2002. - Zitiert auf Seite 17.

[83] V.J. Gillet, P. Willett und J. Bradshaw. **Similarity searching using reduced graphs**. *Journal of Chemical Information and Computer Sciences*, **43**(2):338–345, 2003. doi:10.1021/ci025592e. - Zitiert auf den Seiten 17 und 29.

[84] V.J. Gillet, G.M. Downs, J.D. Holliday, M.F. Lynch und W. Dethlefsen. **Computer storage and retrieval of generic chemical structures in patents. 13. Reduced graph generation**. *Journal of Chemical Information and Modeling*, **31**(2):260–270, 1991. doi:10.1021/ci00002a011. - Zitiert auf Seite 17.

[85] E.J. Barker, E.J. Gardiner, V.J. Gillet, P. Kitts und J. Morris. **Further development of reduced graphs for identifying bioactive compounds**. *Journal of Chemical Information and Computer Sciences*, **43**(2):346–356, 2003. doi:10.1021/ci0255937. - Zitiert auf Seite 17.

[86] Y. Takahashi, M. Sukekawa und S. Sasaki. **Automatic identification of molecular similarity using reduced-graph representation of chemical structure**. *Journal of Chemical Information and Modeling*, **32**(6):639–643, 1992. doi:10.1021/ci00010a009. - Zitiert auf Seite 17.

[87] W. Fisanick, A.H. Lipkus und A. Rusinko. **Similarity searching on CAS Registry substances. 2. 2D structural similarity**. *Journal of Chemical Information and Modeling*, **34**(1):130–140, 1994. doi:10.1021/ci00017a016. - Zitiert auf Seite 17.

[88] G. Harper, G.S. Bravi, S.D. Pickett, J. Hussain und D.V.S. Green. **The Reduced Graph Descriptor in Virtual Screening and Data-Driven Clustering of High-Throughput Screening Data**. *Journal of Chemical Information and Modeling*, **44**(6):2145–2156, 2004. doi:10.1021/ci049860f. - Zitiert auf Seite 17.

[89] M. Boehm, T.-Y. Wu, H. Claussen und C. Lemmen. **Similarity searching and scaffold hopping in synthetically accessible combinatorial chemistry spaces**. *Journal of Medicinal Chemistry*, **51**(8):2468–2480, 2008. doi:10.1021/jm0707727. - Zitiert auf den Seiten 17, 25, 26, 28, 29, 39 und 135.

[90] H.J. Boehm, A. Flohr und M. Stahl. **Scaffold hopping**. *Drug Discovery Today: Technologies*, **1**(3):217–224, 2004. doi:10.1016/j.ddtec.2004.10.009. - Zitiert auf Seite 17.

[91] K. Birchall, V.J. Gillet, G. Harper und S.D. Pickett. **Training Similarity Measures for Specific Activities:**. *Journal of Chemical Information and Modeling*, **46**(2):577–586, 2006. doi:10.1021/ci050465e. - Zitiert auf Seite 17.

[92] R.P. Sheridan und S.K. Kearsley. **Why do we need so many chemical similarity search methods?** *Drug Discovery Today*, **7**(17):903–911, 2002. - Zitiert auf Seite 18.

[93] J.H. Nettles, J.L. Jenkins, A. Bender, Z. Deng, J.W. Davies und M. Glick. **Bridging chemical and biological space: ''target fishing'' using 2D and 3D molecular descriptors**. *Journal of Medicinal Chemistry*, **49**(23):6802–6810, 2006. doi:10.1021/jm060902w. - Zitiert auf Seite 18.

[94] J. Gasteiger und T. Engel, Hrsg. *Chemoinformatics*. Wiley-VCH, 2003. - Zitiert auf Seite 18.

[95] R. Todescini und V. Consonni. *Methods and Principles in Medicinal Chemistry*, **11**, Kapitel Handbook of Molecular Descriptors. Wiley-VCH, 2000. - Zitiert auf Seite 18.

[96] T.I. Oprea. **Property distribution of drug-related chemical databases**. *Journal of Computer-Aided Molecular Design*, **14**(3):251–264, 2000. - Zitiert auf Seite 18.

[97] M.M. Hann, A.R. Leach und G. Harper. **Molecular complexity and its impact on the probability of finding leads for drug discovery**. *Journal of Chemical Information and Computer Sciences*, **41**(3):856–864, 2001. - Zitiert auf Seite 18.

[98] C. Lipinski und A. Hopkins. **Navigating chemical space for biology and medicine**. *Nature*, **432**(7019):855–861, 2004. doi:10.1038/nature03193. - Zitiert auf den Seiten 18 und 25.

[99] H. Kubinyi. **Drug research: myths, hype and reality**. *Nature Reviews. Drug Discovery*, **2**(8):665–668, 2003. doi:10.1038/nrd1156. - Zitiert auf Seite 18.

[100] P. D. Leeson und A. M. Davis. **Drug-like properties: guiding principles for design – or chememical prejudice?** *Drug Discovery Today – Technologies*, **1**(3):189–195, 2004. - Zitiert auf Seite 18.

[101] T. Hou und J. Wang. **Structure-ADME relationship: still a long way to go?** *Expert Opinion on Drug Metabolism & Toxicology*, **4**(6):759–770, 2008. doi:10.1517/17425255.4.6.759. - Zitiert auf Seite 18.

[102] R. Mannhold, G.I. Poda, C. Ostermann und I.V. Tetko. **Calculation of molecular lipophilicity: State-of-the-art and comparison of log P methods on more than 96,000 compounds**. *Journal of Pharmaceutical Sciences*, **98**(3):861–893, 2009. doi:10.1002/jps.21494. - Zitiert auf Seite 18.

[103] M.M. Hann und T.I. Oprea. **Pursuing the leadlikeness concept in pharmaceutical research**. *Current Opinion in Chemical Biology*, **8**(3):255–263, 2004. doi:10.1016/j.cbpa.2004.04.003. - Zitiert auf Seite 19.

[104] T.I. Oprea, A.M. Davis, S.J. Teague und P.D. Leeson. **Is there a difference between leads and drugs? A historical perspective**. *Journal of Chemical Information and Computer Sciences*, **41**(5):1308–1315, 2001. - Zitiert auf den Seiten 19, 87 und 88.

[105] R.A.J. Goodnow, W. Guba und W. Haap. **Library design practices for success in lead generation with small molecule libraries**. *Combinatorial Chemistry & High Throughput Screening*, **6**(7):649–660, 2003. - Zitiert auf Seite 19.

[106] C.A. Lipinski, F. Lombardo, B.W. Dominy und P.J. Feeney. **Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings**. *Advanced Drug Delivery Reviews*, **46**(1-3):3–26, 2001. - Zitiert auf den Seiten 19 und 63.

[107] A.K. Ghose, V.N. Viswanadhan und J.J. Wendoloski. **A Knowledge-Based Approach in Designing Combinatorial or Medicinal Chemistry Libraries for Drug Discovery. 1. A Qualitative and Quantitative Characterization of Known Drug Databases**. *Journal of Combinatorial Chemistry*, **1**(1):55–68, 1999. doi:10.1021/cc9800071. - Zitiert auf Seite 19.

[108] S.J. Teague, A.M. Davis, P.D. Leeson und T. Oprea. **The Design of Leadlike Combinatorial Libraries**. *Angewandte Chemie (International ed. in English)*, **38**(24):3743–3748, 1999. - Zitiert auf Seite 19.

[109] M. Congreve, R. Carr, C. Murray und H. Jhoti. **A 'rule of three' for fragment-based lead discovery?** *Drug Discovery Today*, **8**(19):876–877, 2003. - Zitiert auf Seite 19.

[110] U. Norinder und M. Haeberlein. **Computational approaches to the prediction of the blood-brain distribution**. *Advanced Drug Delivery Reviews*, **54**(3):291–313, 2002. - Zitiert auf Seite 19.

[111] T.W. von Geldern, D.J. Hoffman, J.A. Kester, H.N. Nellans, B.D. Dayton, S.V. Calzadilla, K.C. Marsh, L. Hernandez, W. Chiou, D.B. Dixon, J.R. Wu-Wong und T.J. Opgenorth. **Azole endothelin antagonists. 3. Using delta log P as a tool to improve absorption**. *Journal of Medicinal Chemistry*, **39**(4):982–991, 1996. doi:10.1021/jm9505932. - Zitiert auf Seite 19.

[112] K. Palm, P. Stenberg, K. Luthman und P. Artursson. **Polar molecular surface properties predict the intestinal absorption of drugs in humans**. *Pharmaceutical Research*, **14**(5):568–571, 1997. - Zitiert auf Seite 19.

[113] J. Kelder, P.D. Grootenhuis, D.M. Bayada, L.P. Delbressine und J.P. Ploemen. **Polar molecular surface as a dominating determinant for oral absorption and brain penetration of drugs**. *Pharmaceutical Research*, **16**(10):1514–1519, 1999. - Zitiert auf Seite 19.

[114] P. Ertl, B. Rohde und P. Selzer. **Fast calculation of molecular polar surface area as a sum of fragment-based contributions and its application to the prediction of drug transport properties**. *Journal of Medicinal Chemistry*, **43**(20):3714–3717, 2000. - Zitiert auf den Seiten 19, 42, 46 und 60.

[115] D.F. Veber, S.R. Johnson, H.-Y. Cheng, B.R. Smith, K.W. Ward und K.D. Kopple. **Molecular properties that influence the oral bioavailability of drug candidates**. *Journal of Medicinal Chemistry*, **45**(12):2615–2623, 2002. - Zitiert auf Seite 19.

[116] S.D. Pickett, I.M. McLay und D.E. Clark. **Enhancing the hit-to-lead properties of lead optimization libraries**. *Journal of Chemical Information and Computer Sciences*, **40**(2):263–272, 2000. - Zitiert auf Seite 19.

[117] M. Rarey, B. Kramer, T. Lengauer und G. Klebe. **A fast flexible docking method using an incremental construction algorithm**. *Journal of Molecular Biology*, **261**(3):470–489, 1996. doi:10.1006/jmbi.1996.0477. - Zitiert auf den Seiten 20, 30, 31 und 46.

[118] M. Rarey und T. Lengauer. **A recursive algorithm for efficient combinatorial library docking**. *Perspectives in Drug Discovery and Design*, **20**(1):63–81, 2000. doi:10.1023/A:1008716720979. - Zitiert auf Seite 20.

[119] V. Gillet und P. Willett. **Dissimilarity-Based Compound Selection For Library Design**. In *Combinatorial Library Design and Evaluation: Principles, Software, Tools, and Applications in Drug Discovery*, Kapitel 13. CRC Press, 2001. - Zitiert auf Seite 20.

[120] J.H. Ward. **Hierarchical Grouping to Optimize an Objective Function**. *Journal of the American Statistical Association*, **58**(301):236, 1963. doi:10.2307/2282967. - Zitiert auf den Seiten 21 und 123.

[121] J. MacQueen. **Some methods for classification and analysis of multivariate observations**. *Proc. Fifth Berkeley Sympos. Math. Statist. and Probability (Berkeley, Calif., 1965/66), Vol. I: Statistics*, S. 281âĂŞ297, 1967. - Zitiert auf Seite 21.

[122] L. Kaufman und P. Rousseeuw. *Clustering by means of medoids*. Faculty of Mathematics and Informatics, Delft, 1987. - Zitiert auf den Seiten 21 und 123.

[123] R.A. Jarvis und E.A. Patrick. **Clustering Using a Similarity Measure Based on Shared Near Neighbors**. *IEEE Transactions on Computers*, **C-22**(11):1025–1034, 1973. doi:10.1109/T-C.1973.223640. - Zitiert auf den Seiten 21 und 123.

[124] A.K. Jain, M.N. Murty und P.J. Flynn. **Data clustering: a review**. *ACM Computing Surveys*, **31**(3):264–323, 1999. doi:10.1145/331499.331504. - Zitiert auf Seite 21.

[125] Geoff M. Downs und John M. Barnard. **Clustering Methods and Their Uses in Computational Chemistry**. In Kenny B. Lipkowitz und Donald B. Boyd, Hrsg., *Reviews in Computational Chemistry, Volume 18*. Wiley-VCH, 2002. doi:10.1002/0471433519.ch1. - Zitiert auf Seite 21.

[126] K.V. Balakin, A.V. Kozintsev, A.S. Kiselyov und N.P. Savchuk. **Rational design approaches to chemical libraries for hit identification**. *Current Drug Discovery Technologies*, **3**(1):49–65, 2006. - Zitiert auf Seite 21.

[127] A.R. Leach und M.M. Hann. **The in silico world of virtual libraries**. *Drug Discovery Today*, **5**(8):326–336, 2000. - Zitiert auf Seite 22.

[128] A.R. Leach, R.A. Bryce und A.J. Robinson. **Synergy between combinatorial chemistry and de novo design**. *Journal of Molecular Graphics & Modelling*, **18**(4-5):358–67, 526, 2000. - Zitiert auf Seite 22.

[129] R.D. Brown und Y.C. Martin. **Designing combinatorial library mixtures using a genetic algorithm**. *Journal of Medicinal Chemistry*, **40**(15):2304–2313, 1997. doi:10.1021/jm970033y. - Zitiert auf Seite 22.

[130] G. Bravi, D.V. Green, M.M. Hann und A.R. Leach. **PLUMS: a program for the rapid optimization of focused libraries**. *Journal of Chemical Information and Computer Sciences*, **40**(6):1441–1448, 2000. - Zitiert auf Seite 22.

[131] W. Zheng, S.T. Hung, J.T. Saunders und G.L. Seibel. **PICCOLO: a tool for combinatorial library design via multicriterion optimization**. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, S. 588–599, 2000. - Zitiert auf Seite 22.

[132] V.J. Gillet, P. Willett, J. Bradshaw und D.V.S. Green. **Selecting Combinatorial Libraries to Optimize Diversity and Physical Properties**. *Journal of Chemical Information and Modeling*, **39**(1):169–177, 1999. doi:10.1021/ci980332b. - Zitiert auf Seite 23.

[133] T. Wright, V.J. Gillet, D.V.S. Green und S.D. Pickett. **Optimizing the size and configuration of combinatorial libraries**. *Journal of Chemical Information and Computer Sciences*, **43**(2):381–390, 2003. doi:10.1021/ci0255836. - Zitiert auf Seite 23.

[134] G. Chen, S. Zheng, X. Luo, J. Shen, W. Zhu, H. Liu, C. Gui, J. Zhang, M. Zheng, C.M. Puah, K. Chen und H. Jiang. **Focused combinatorial library design based on structural diversity, druglikeness and binding affinity score**. *Journal of Combinatorial Chemistry*, **7**(3):398–406, 2005. doi:10.1021/cc049866h. - Zitiert auf Seite 23.

[135] T.J. Ewing, S. Makino, A.G. Skillman und I.D. Kuntz. **DOCK 4.0: search strategies for automated molecular docking of flexible molecule databases**. *Journal of Computer-Aided Molecular Design*, **15**(5):411–428, 2001. - Zitiert auf Seite 23.

[136] C. Le Bailly de Tilleghem, B. Beck, B. Boulanger und B. Govaerts. **A fast exchange algorithm for designing focused libraries in lead optimization**. *Journal of Chemical Information and Modeling*, **45**(3):758–767, 2005. doi:10.1021/ci049787t. - Zitiert auf den Seiten 23 und 55.

[137] M. Hartenfeller, E. Proschak, A. Schueller und G. Schneider. **Concept of combinatorial de novo design of drug-like molecules by particle swarm optimization**. *Chemical Biology & Drug Design*, **72**(1):16–26, 2008. doi:10.1111/j.1747-0285.2008.00672.x. - Zitiert auf den Seiten 23 und 26.

[138] G. Schneider, W. Neidhart, T. Giller und G. Schmid. **Scaffold-Hopping by Topological Pharmacophore Search: A Contribution to Virtual Screening**. *Angewandte Chemie (International ed. in English)*, **38**(19):2894–2896, 1999. - Zitiert auf Seite 23.

[139] P. Pfeffer, T. Fober, E. Hullermeier und G. Klebe. **GARLig: a fully automated tool for subset selection of large fragment spaces via a self-adaptive genetic algorithm**. *Journal of Chemical Information and Modeling*, **50**(9):1644–1659, 2010. doi:10.1021/ci9003305. - Zitiert auf Seite 23.

[140] D.S. Goodsell und A.J. Olson. **Automated docking of substrates to proteins by simulated annealing**. *Proteins*, **8**(3):195–202, 1990. doi:10.1002/prot.340080302. - Zitiert auf Seite 23.

[141] **AutoDock** [online]. 2011. The Scrips Research Institute, La Jolla, CA. http://autodock.scripps.edu/ [Abruf: 22.06.2011]. - Zitiert auf Seite 23.

[142] G. Jones, P. Willett und R.C. Glen. **A genetic algorithm for flexible molecular overlay and pharmacophore elucidation**. *Journal of Computer-Aided Molecular Design*, **9**(6):532–549, 1995. - Zitiert auf Seite 23.

[143] **GOLD** [online]. 2010. The Cambridge Crystallographic Data Centre, Cambridge, UK. http://www.ccdc.cam.ac.uk/products/life_sciences/gold/ [Abruf: 22.03.2011]. - Zitiert auf Seite 23.

[144] **Chil2** [online]. http://www.chil2.de/index.html [Abruf: 22.03.2011]. - Zitiert auf Seite 24.

[145] U. Lessel, B. Wellenzohn, M. Lilienthal und H. Claussen. **Searching Fragment Spaces with Feature Trees**. *Journal of Chemical Information and Modeling*, **49**(2):270–279, 2009. doi:10.1021/ci800272a. - Zitiert auf den Seiten 25, 28 und 39.

[146] P.S. Kutchukian und E.I. Shakhnovich. **De novo design: balancing novelty and confined chemical space**. *Expert Opinion on Drug Discovery*, **5**(8):789–812, 2010. doi:10.1517/17460441.2010.497534. - Zitiert auf den Seiten 25 und 26.

[147] C.M. Dobson. **Chemical space and biology**. *Nature*, **432**(7019):824–828, 2004. doi:10.1038/nature03192. - Zitiert auf Seite 25.

[148] T. Fink, H. Bruggesser und J.-L. Reymond. **Virtual exploration of the small-molecule chemical universe below 160 Daltons**. *Angewandte Chemie (International ed. in English)*, **44**(10):1504–1508, 2005. doi:10.1002/anie.200462457. - Zitiert auf Seite 25.

[149] R. van Deursen und J.-L. Reymond. **Chemical space travel**. *ChemMedChem*, **2**(5):636–640, 2007. doi:10.1002/cmdc.200700021. - Zitiert auf Seite 25.

[150] T. Fink und J.-L. Reymond. **Virtual exploration of the chemical universe up to 11 atoms of C, N, O, F: assembly of 26.4 million structures (110.9 million stereoisomers) and analysis for new ring systems, stereochemistry, physicochemical properties, compound classes, and drug discovery**. *Journal of Chemical Information and Modeling*, **47**(2):342–353, 2007. doi:10.1021/ci600423u. - Zitiert auf Seite 25.

[151] G. Schneider und U. Fechner. **Computer-based de novo design of drug-like molecules**. *Nature Reviews. Drug Discovery*, **4**(8):649–663, 2005. doi:10.1038/nrd1799. - Zitiert auf Seite 26.

[152] M. Hartenfeller und G. Schneider. **De novo drug design**. *Methods in Molecular Biology (Totowa, NJ, United States)*, **672**:299–323, 2011. doi:10.1007/978-1-60761-839-3_12. - Zitiert auf Seite 26.

[153] H. Mauser und W. Guba. **Recent developments in de novo design and scaffold hopping**. *Current Opinion in Drug Discovery & Development*, **11**(3):365–374, 2008. - Zitiert auf Seite 26.

[154] P. Maass, T. Schulz-Gasch, M. Stahl und M. Rarey. **Recore: a fast and versatile method for scaffold hopping based on small molecule crystal structure conformations**. *Journal of Chemical Information and Modeling*, **47**(2):390–399, 2007. doi:10.1021/ci060094h. - Zitiert auf Seite 26.

[155] G. Schneider, M.L. Lee, M. Stahl und P. Schneider. **De novo design of molecular architectures by evolutionary assembly of drug-derived building blocks**. *Journal of Computer-Aided Molecular Design*, **14**(5):487–494, 2000. - Zitiert auf Seite 26.

[156] U. Fechner und G. Schneider. **Flux (1): a virtual synthesis scheme for fragment-based de novo design**. *Journal of Chemical Information and Modeling*, **46**(2):699–707, 2006. doi:10.1021/ci0503560. - Zitiert auf Seite 26.

[157] U. Fechner und G. Schneider. **Flux (2): comparison of molecular mutation and crossover operators for ligand-based de novo design**. *Journal of Chemical Information and Modeling*, **47**(2):656–667, 2007. doi:10.1021/ci6005307. - Zitiert auf Seite 26.

[158] T. Lippert, T. Schulz-Gasch, O. Roche, W. Guba und M. Rarey. **De novo design by pharmacophore-based searches in fragment spaces**. *Journal of Computer-Aided Molecular Design*, **25**(10):931–945, 2011. doi:10.1007/s10822-011-9473-6. - Zitiert auf den Seiten 26 und 64.

[159] A. Zaliani, K. Boda, T. Seidel, A. Herwig, C.H. Schwab, J. Gasteiger, H. Claussen, C. Lemmen, J. Degen, J. Parn und M. Rarey. **Second-generation de novo design: a view from a medicinal chemist perspective**. *Journal of Computer-Aided Molecular Design*, 2009. doi:10.1007/s10822-009-9291-2. - Zitiert auf Seite 26.

[160] X.Q. Lewell, D.B. Judd, S.P. Watson und M.M. Hann. **RECAP–retrosynthetic combinatorial analysis procedure: a powerful new technique for identifying privileged molecular fragments with useful applications in combinatorial chemistry**. *Journal of Chemical Information and Computer Sciences*, **38**(3):511–522, 1998. - Zitiert auf den Seiten 26 und 27.

[161] Thomson Reuters. **World Drug Index (WDI)** [online]. http://thomsonreuters.com/products_services/science/science_products/a-z/world_drug_index/. - Zitiert auf den Seiten 26 und 80.

[162] H. Mauser und M. Stahl. **Chemical fragment spaces for de novo design**. *Journal of Chemical Information and Modeling*, **47**(2):318–324, 2007. doi:10.1021/ci6003652. - Zitiert auf Seite 27.

[163] J. Degen, C. Wegscheid-Gerlach, A. Zaliani und M. Rarey. **On the art of compiling and using 'drug-like' chemical fragment spaces**. *ChemMedChem*, **3**(10):1503–1507, 2008. doi:10.1002/cmdc.200800178. - Zitiert auf den Seiten 27 und 80.

[164] M. Rarey. **Computergestuetzes Wirkstoffdesign mit chemischen Fragmentraeumen**. *BIOforum*, **10**:56–59, 2005. - Zitiert auf Seite 27.

[165] M. Rarey, S. Hindle, P. Maass, G. Metz, C. Rummey und M. Zimmermann. *Pharmacophores and Pharmacophore Search*, **32** of *Methods and Principles in Medicinal Chemistry*, Kapitel Feature Trees: Theory and Applications from Large-Scale Virtual Screening to Data Analysis, S. 81–116. Wiley-VCH, Weinheim, 2005. - Zitiert auf Seite 29.

[166] J.R. Fischer und M. Rarey. **SwiFT: an index structure for reduced graph descriptors in virtual screening and clustering**. *Journal of Chemical Information and Modeling*, **47**(4):1341–1353, 2007. doi:10.1021/ci700007b. - Zitiert auf den Seiten 29, 36, 66, 83, 125 und 126.

[167] D.R. Flower. **On the Properties of Bit String-Based Measures of Chemical Similarity**. *Journal of Chemical Information and Modeling*, **38**(3):379–386, 1998. doi:10.1021/ci970437z. - Zitiert auf Seite 29.

[168] T. H. Cormen, C. E. Leiserson und R. L. Rivest. *Algorithms*. MIT Press, 1998. - Zitiert auf den Seiten 30, 33 und 36.

[169] T. Ottmann und P.Widmayer. *Algorithmen und Datenstrukturen*. Spektrum Akademischer Verlag, 2002. - Zitiert auf Seite 33.

[170] M. Zimmermann. *Rechnergestützte Analyse von HTS-Daten*. Dissertation, Rheinische Friedrich-Wilhelms-Universität Bonn, 2003. - Zitiert auf Seite 34.

[171] G. Hessler, M. Zimmermann, H. Matter, A. Evers, T. Naumann, T. Lengauer und M. Rarey. **Multiple-ligand-based virtual screening: methods and applications of the MTree approach**. *Journal of Medicinal Chemistry*, **48**(21):6575–6584, 2005. doi:10.1021/jm050078w. - Zitiert auf Seite 34.

[172] R. Bellman. **On the Theory of Dynamic Programming**. *Proceedings of the National Academy of Sciences*, **38**(8):716–719, 1952. doi:10.1073/pnas.38.8.716. - Zitiert auf Seite 36.

LITERATURVERZEICHNIS

[173] D. Weininger, A. Weininger und J.L. Weininger. **SMILES. 2. Algorithm for generation of unique SMILES notation**. *Journal of Chemical Information and Computer Sciences*, **29**(2):97–101, 1989. doi:10.1021/ci00062a008. - Zitiert auf den Seiten 41, 45, 136 und 138.

[174] SYMYX. **CTfile Formats** [online]. http://www.symyx.com/downloads/public/ctfile/ctfile.jsp [Abruf: 27.01.2011]. - Zitiert auf den Seiten 41, 136 und 138.

[175] **TRIPOS Mol2 File Format** [online]. http://tripos.com/data/support/mol2.pdf [Abruf: 27.01.2011]. - Zitiert auf den Seiten 41, 81, 136 und 138.

[176] J.R. Ullmann. **An algorithm for subgraph isomorphism**. *Journal of the Association for Computing Machinery*, **23**:31–42, 1976. - Zitiert auf Seite 41.

[177] L.P. Cordella, P. Foggia, C. Sansone und M. Vento. **A (sub)graph isomorphism algorithm for matching large graphs**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **26**(10):1367–1372, 2004. doi:10.1109/TPAMI.2004.75. - Zitiert auf Seite 41.

[178] A.K. Ghose und G.M. Crippen. **Atomic physicochemical parameters for three-dimensional-structure-directed quantitative structure-activity relationships. 2. Modeling dispersive and hydrophobic interactions**. *Journal of Chemical Information and Computer Sciences*, **27**(1):21–35, 1987. - Zitiert auf Seite 42.

[179] S.A. Wildman und G.M. Crippen. **Prediction of Physicochemical Parameters by Atomic Contributions**. *Journal of Chemical Information and Modeling*, **39**(5):868–873, 1999. doi:10.1021/ci9903071. - Zitiert auf Seite 42.

[180] J. Braun, R. Gugisch, A. Kerber, R. Laue, M. Meringer und C. Rucker. **MOLGEN-CID–A canonizer for molecules and graphs accessible through the Internet**. *Journal of Chemical Information and Computer Sciences*, **44**(2):542–548, 2004. doi:10.1021/ci0304041. - Zitiert auf Seite 45.

[181] Ronald J. Gillespie und Istvan Hargittai. **The Vsepr Model of Molecular Geometry**. 1991. - Zitiert auf Seite 46.

[182] W.H. Powell. **Revision of the extended Hantzsch-Widman system of nomenclature for heteromonocycles**. *Pure and Applied Chemistry*, **55**(2):409–416, 1983. doi:10.1351/pac198855020409. - Zitiert auf Seite 47.

[183] R.A. Ward und J.G. Kettle. **Systematic Enumeration of Heteroaromatic Ring Systems as Reagents for Use in Medicinal Chemistry**. *Journal of Medicinal Chemistry*, **54**(13):4670–4677, 2011. doi:10.1021/jm200338a. - Zitiert auf Seite 47.

[184] G. Dueck und T. Scheuer. **Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing**. *Journal of Computational Physics*, **90**(1):161–175, 1990. doi:10.1016/0021-9991(90)90201-B. - Zitiert auf Seite 53.

[185] G. Dueck. **New Optimization Heuristics The Great Deluge Algorithm and the Record-to-Record Travel**. *Journal of Computational Physics*, **104**(1):86–92, 1993. doi:10.1006/jcph.1993.1010. - Zitiert auf Seite 53.

[186] R. Eglese. **Simulated annealing: A tool for operational research**. *European Journal of Operational Research*, **46**(3):271–281, 1990. doi:10.1016/0377-2217(90)90001-R. - Zitiert auf Seite 54.

[187] R.D. Brown, M. Hassan und M. Waldman. **Combinatorial library design for diversity, cost efficiency, and drug-like character**. *Journal of Molecular Graphics & Modelling*, **18**(4-5):427–37, 537, 2000. - Zitiert auf Seite 55.

[188] P. Vismara. **Union of all the minimum cycle bases of a graph**. *The electronic journal of combinatorics*, **4**:1–15, 1997. - Zitiert auf Seite 61.

[189] U. Norinder und C.A.S. Bergstrom. **Prediction of ADMET Properties**. *ChemMedChem*, **1**(9):920–937, 2006. doi:10.1002/cmdc.200600155. - Zitiert auf Seite 62.

[190] BioSolveIT, St. Augustin, Deutschland. **CoLibri** [online]. http://www.biosolveit.de/Colibri [Abruf: 22.01.2011]. - Zitiert auf den Seiten 64 und 135.

[191] P.C. Fricker, M. Gastreich und M. Rarey. **Automated drawing of structural molecular formulas under constraints**. *Journal of Chemical Information and Computer Sciences*, **44**(3):1065–1078, 2004. doi:10.1021/ci049958u. - Zitiert auf den Seiten 71, 138 und 145.

[192] R. Aslanian, M.W. Mutahi, N.Y. Shih, K.D. McCormick, J.J. Piwinski, P.C. Ting, M.M. Albanese, M.Y. Berlin, X. Zhu, S.C. Wong, S.B. Rosenblum, Y. Jiang, R. West, S. She, S.M. Williams, M. Bryant und J.A. Hey. **Identification of a novel, orally bioavailable histamine H(3) receptor antagonist based on the 4-benzyl-(1H-imidazol-4-yl) template**. *Bioorganic & Medicinal Chemistry Letters*, **12**(6):937–941, 2002. - Zitiert auf den Seiten 71, 91 und 95.

[193] S. Celanire, M. Wijtmans, P. Talaga, R. Leurs und I.J.P. de Esch. **Keynote review: histamine H3 receptor antagonists reach out for the clinic**. *Drug Discovery Today*, **10**(23-24):1613–1627, 2005. doi:10.1016/S1359-6446(05)03625-1. - Zitiert auf den Seiten 71 und 91.

[194] C. Lemmen, T. Lengauer und G. Klebe. **FLEXS: a method for fast flexible ligand superposition**. *Journal of Medicinal Chemistry*, **41**(23):4502–4520, 1998. doi:10.1021/jm9810371. - Zitiert auf den Seiten 77, 109, 122, 123 und 138.

[195] BioSolveIT, St. Augustin, Deutschland. **FlexS** [online]. http://www.biosolveit.de/FlexS [Abruf: 22.01.2011]. - Zitiert auf Seite 77.

[196] T.S. Rush, J.A. Grant, L. Mosyak und A. Nicholls. **A shape-based 3-D scaffold hopping method and its application to a bacterial protein-protein interaction**. *Journal of Medicinal Chemistry*, **48**(5):1489–1495, 2005. doi:10.1021/jm040163o. - Zitiert auf Seite 77.

[197] OpenEye Scientific Software, Santa Fe, NM. **ROCS** [online]. 2010. http://http://www.eyesopen.com. - Zitiert auf den Seiten 77 und 100.

[198] Novell. **openSUSE 11.1** [online]. http://www.opensuse.org [Abruf: 02.01.2011]. - Zitiert auf Seite 79.

[199] A.G. Cole, I.L. Stroke, M.-R. Brescia, S. Simhadri, J.J. Zhang, Z. Hussain, M. Snider, C. Haskell, S. Ribeiro, K.C. Appell, I. Henderson und M.L. Webb. **Identification and initial evaluation of 4-N-aryl-[1,4]diazepane ureas as potent CXCR3 antagonists**. *Bioorganic & Medicinal Chemistry Letters*, **16**(1):200–203, 2006. doi:10.1016/j.bmcl.2005.09.020. - Zitiert auf Seite 79.

[200] M. Wijtmans, D. Verzijl, R. Leurs, I.J.P. de Esch und M.J. Smit. **Towards small-molecule CXCR3 ligands with clinical potential**. *ChemMedChem*, **3**(6):861–872, 2008. doi:10.1002/cmdc.200700365. - Zitiert auf Seite 79.

[201] J.J. Irwin und B.K. Shoichet. **ZINC–a free database of commercially available compounds for virtual screening**. *Journal of Chemical Information and Modeling*, **45**(1):177–182, 2005. doi:10.1021/ci049714+. - Zitiert auf Seite 80.

[202] N. Huang, B.K. Shoichet und J.J. Irwin. **Benchmarking sets for molecular docking**. *Journal of Medicinal Chemistry*, **49**(23):6789–6801, 2006. doi:10.1021/jm0608356. - Zitiert auf den Seiten 82, 102, 125 und 128.

[203] J. Hert, P. Willett, D.J. Wilton, P. Acklin, K. Azzaoui, E. Jacoby und A. Schuffenhauer. **Comparison of fingerprint-based methods for virtual screening using multiple bioactive reference structures**. *Journal of Chemical Information and Computer Sciences*, **44**(3):1177–1185, 2004. doi:10.1021/ci034231b. - Zitiert auf den Seiten 82, 125 und 126.

[204] MDL. **MDL Drug Data Report** [online]. http://www.mdli.com [Abruf: 07.01.2011]. - Zitiert auf den Seiten 82 und 126.

[205] BioSolveIT, St. Augustin, Deutschland. **Feature Trees** [online]. http://www.biosolveit.de/FTrees [Abruf: 22.01.2011]. - Zitiert auf den Seiten 84 und 144.

[206] BioSolveIT, St. Augustin, Deutschland. **Knowledge Space** [online]. http://www.biosolveit.de/datasets/#knowledgespace [Abruf: 07.02.2011]. - Zitiert auf Seite 86.

[207] J.-M. Arrang, M. Garbarg und J.-C. Schwartz. **Auto-inhibition of brain histamine release mediated by a novel class (H3) of histamine receptor**. *Nature*, **302**(5911):832–837, 1983. doi:10.1038/302832a0. - Zitiert auf Seite 91.

[208] T.A. Esbenshade, G.B. Fox, K.M. Krueger, T.R. Miller, C.H. Kang, L.I. Denny, D.G. Witte, B.B. Yao, L. Pan, J. Wetter, K. Marsh, Y.L. Bennani, M.D. Cowart, J.P. Sullivan und A.A. Hancock. **Pharmacological properties of ABT-239 [4-(2-{2-[(2R)-2-Methylpyrrolidinyl]ethyl}-benzofuran-5-yl)benzonitrile]: I. Potent and selective histamine H3 receptor antagonist with drug-like properties**. *The Journal of Pharmacology and Experimental Therapeutics*, **313**(1):165–175, 2005. doi:10.1124/jpet.104.078303. - Zitiert auf Seite 91.

[209] J.F. Lau, C.B. Jeppesen, K. Rimvall und R. Hohlweg. **Ureas with histamine H3-antagonist receptor activity–a new scaffold discovered by lead-hopping from cinnamic acid amides**. *Bioorganic & Medicinal Chemistry Letters*, **16**(20):5303–5308, 2006. doi:10.1016/j.bmcl.2006.07.093. - Zitiert auf Seite 91.

[210] R. Aslanian, J.E. Brown, N.Y. Shih, M. wa Mutahi, M.J. Green, S. She, M. Del Prado, R. West und J. Hey. **4-[(1H-imidazol-4-yl) methyl] benzamidines and benzylamidines: novel antagonists of the histamine H3 receptor**. *Bioorganic & Medicinal Chemistry Letters*, **8**(16):2263–2268, 1998. - Zitiert auf Seite 95.

[211] E.A. Nigg. **Cyclin-dependent protein kinases: key regulators of the eukaryotic cell cycle**. *BioEssays : News and Reviews in Molecular, Cellular and Developmental Biology*, **17**(6):471–480, 1995. doi:10.1002/bies.950170603. - Zitiert auf Seite 96.

[212] S. Wadler. **Perspectives for cancer therapies with cdk2 inhibitors**. *Drug Resistance Updates : Reviews and Commentaries in Antimicrobial and Anticancer Chemotherapy*, **4**(6):347–367, 2001. doi:10.1054/drup.2001.0224. - Zitiert auf Seite 96.

[213] T.G. Davies, P. Tunnah, L. Meijer, D. Marko, G. Eisenbrand, J.A. Endicott und M.E. Noble. **Inhibitor binding to active and inactive CDK2: the crystal structure of CDK2-cyclin A/indirubin-5-sulphonate**. *Structure (London, England : 1993)*, **9**(5):389–397, 2001. - Zitiert auf Seite 96.

[214] **MOE** [online]. 2010. Chemical Computing Group: Quebec, Canada. - Zitiert auf Seite 100.

[215] C.J. Swain, A. Teran, M. Maroto und A. Cabello. **Identification and optimisation of 5-amino-7-aryldihydro-1,4-diazepines as 5-HT2A ligands**. *Bioorganic & Medicinal Chemistry Letters*, **16**(23):6058–6062, 2006. doi:10.1016/j.bmcl.2006.08.108. - Zitiert auf Seite 108.

[216] A.L. Smith, G.I. Stevenson, S. Lewis, S. Patel und J.L. Castro. **Solid-phase synthesis of 2,3-disubstituted indoles: discovery of a novel, high-affinity, selective h5-HT2A antagonist**. *Bioorganic & Medicinal Chemistry Letters*, **10**(24):2693–2696, 2000. - Zitiert auf Seite 108.

[217] J.W. Raymond, E.J. Gardiner und P. Willett. **Heuristics for similarity searching of chemical graphs using a maximum common edge subgraph algorithm**. *Journal of Chemical Information and Computer Sciences*, **42**(2):305–316, 2002. - Zitiert auf Seite 123.

[218] J.W. Raymond, C.J. Blankley und P. Willett. **Comparison of chemical clustering methods using graph- and fingerprint-based similarity measures**. *Journal of Molecular Graphics & Modelling*, **21**(5):421–433, 2003. - Zitiert auf Seite 123.

LITERATURVERZEICHNIS

[219] M. Ankerst, M.M. Breunig, H.-P. Kriegel und J. Sander. **OPTICS: ordering points to identify the clustering structure**. *ACM SIGMOD Record*, **28**(2):49–60, 1999. doi:10.1145/304181.304187. - Zitiert auf Seite 123.

[220] Y. Loewenstein, E. Portugaly, M. Fromer und M. Linial. **Efficient algorithms for accurate hierarchical clustering of huge datasets: tackling the entire protein space**. *Bioinformatics*, **24**(13):i41–i49, 2008. doi:10.1093/bioinformatics/btn174. - Zitiert auf Seite 123.

[221] **SWIG** [online]. http://swig.org [Abruf: 02.01.2011]. - Zitiert auf Seite 126.

[222] BioSolveIT, St. Augustin, Deutschland. **FlexV** [online]. http://www.biosolveit.de/FlexV [Abruf: 07.01.2011]. - Zitiert auf Seite 137.

[223] TIBCO Software Inc, Palo Alto, CA. **Spotfire** [online]. http://spotfire.tibco.com/ [Abruf: 22.01.2011]. - Zitiert auf Seite 138.

[224] **Standard Template Library (STL)** [online]. http://www.cplusplus.com/reference/stl/ [Abruf: 02.01.2011]. - Zitiert auf Seite 144.

[225] **Boost C++** [online]. http://www.boost.org [Abruf: 02.01.2011]. - Zitiert auf den Seiten 144 und 145.

[226] Nokia. **Qt** [online]. http://qt.nokia.com [Abruf: 02.01.2011]. - Zitiert auf Seite 144.

[227] BioSolveIT, St. Augustin, Deutschland. **LeadIT** [online]. http://www.biosolveit.de/LeadIT [Abruf: 22.01.2011]. - Zitiert auf Seite 144.

[228] D. Weininger. **SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules**. *Journal of Chemical Information and Computer Sciences*, **28**(1):31–36, 1988. doi:10.1021/ci00057a005. - Zitiert auf Seite 144.

[229] E. Gamma, R.Helm, R. Johnson und J. M. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, 1994. - Zitiert auf den Seiten 144 und 148.

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form in einem Verfahren zur Erlangung eines akademischen Grades vorgelegt. Es wurde an keinem anderen Fachbereich ein Antrag auf Eröffnung eines Promotionsverfahrens gestellt.

Berlin, den 07. Dezember 2011

(Robert Fischer)