Searching for Generic Chemical Patterns in Combinatorial Chemical Spaces

Dissertation

zur Erlangung des akademischen Grades

Dr. rer. nat.

an der Fakultät für Mathematik, Informatik und Naturwissenschaften

der Universiät Hamburg

eingereicht beim Fach-Promotionsausschuss Informatik von

Hans-Christian Ehrlich

aus Berlin

Februar 2013

Erstgutachter:	Prof. Dr. Matthias Rarey
Zweitgutachter:	Prof. Dr. Wolfgang Menzel

Tag der Disputation: 28. August 2013

Zusammenfassung

Bestehende Ansätze einen chemischen Raum nach Molekülen mit vordefinierten Strukturelementen zu durchsuchen, repräsentieren diesen Raum als eine Menge von expliziten Molekülen. Diese Moleküldatenbanken werden sequenziell oder parallel durchsucht, wobei jedes Molekül auf die Anwesenheit der Teilstruktur getestet wird. Leider ist der so durchsuchte chemische Raum begrenzt durch die Speicherkapazitäten heutiger Datenbanken. Um diese, auch zukünftig bestehende, Begrenzung zu umgehen, wurden in den letzten Jahrzehnten alternative Speicherkonzepte entwickelt, so genannte Fragmenträume. Diese Räume bestehen aus molekularen Fragmenten und Regeln, wie diese Fragmente zu Produkten oder Molekülen verknüpft werden können. Eine so geartete kombinatorische Beschreibung erlaubt es große chemische Räume mit wenigen Fragmenten und Regeln darzustellen. Leider sind bestehende Methoden auf eine solche Beschreibung in der Regel nicht anwendbar und müssen unter großem Aufwand angepasst werden.

In dieser Arbeit wird ein Verfahren zur Suche nach Molekülen mit vordefinierten chemischen Mustern in Fragmenträumen vorgestellt. Das Verfahren basiert auf einem *Teilen und Herrschen* Ansatz, der Fragmente und Verknüpfungsregeln direkt verarbeitet und somit die kostenintensive Aufzählung von kompletten Molekülen vermeidet. Das Ergebnis ist eine vollständige, minimale Menge von Fragmenten, die nach Verknüpfung das gesuchte Muster enthalten. Eine solche Suche nach Molekülen ist von zentralem Interesse in der frühen Phase des Medikamentenentwurfes.

Die hier präsentierte Methode hat sich in mehreren ausgewählten Szenarien, die Prozessen aus dem Wirkstoffentwurf nachempfunden sind, als hilfreich und zuverlässig erwiesen.

Abstract

Standard approaches to search molecules for user-defined chemical patterns scan huge databases in which the chemical space is represented as a set of explicitly stored molecular structures. Regardless of the employed search algorithm, the storage capacities of modern database systems limit the covered chemical space. Alternative storage concepts such as fragment spaces emerged over the last decades to circumvent this limitation. These spaces consist of molecular fragments and rules that describe their possible connections. The combinatorial nature of fragment spaces allows the description of large chemical spaces with only a few fragments and connection rules. Unfortunately, existing methods need major modifications to work on these spaces.

This thesis presents a novel method to search for chemical patterns in fragment spaces. It is based on an algorithm that uses a *divide and conquer* strategy to directly process fragments and connection rules to avoid the costly enumeration of molecules encoded by the fragment space. As a result, the method produces a complete, minimal set of fragments that includes the user-defined pattern after their connection to molecules. The method is of major interest during the early stage of drug discovery. This is demonstrated by conducting multiple tests designed to mimic real world drug discovery scenarios.

Acknowledgment

I like to thank my research advisor Professor Dr. Matthias Rarey for bringing me to the University of Hamburg and giving me the great opportunity to work on this project. His advice on complicated algorithmic problems, his notion for hot topics and the open door policy resulted in an exceptional guidance during my work.

Many thanks to the research team at the Center of Bioinformatics (ZBH) in Hamburg. I appreciated the good working atmosphere and had a great time. Especially, I like to thank Adrian Kolodzik for being a great office co-work. The discussions within and outside our working field were fruitful in many perspectives. Thanks to Jochen Schlosser for guiding me during my early teaching experiences and Angela Henzeler and Matthias Hilbig for the collaboration during the following teaching courses. Addition thanks to Angela Henzler for proof-reading almost every manuscript I wrote and Matthias Hilbig for all the technical advise. Sincere thanks to Florian Lauk, Tim Harder, Lennart Heinzerling, Karen Schomburg, Stefan Bietz and Clemens Kühn for proof-reading this thesis. Over the last four years, many people have left and joined the ZBH: Patrick Maaß, Juri Pärn, Axel Griewel, Björn Windshügel, Birte Seebeck, Robert Fischer, Tobias Lippert, Katrin Stierand, Andrea Volkamer, Sascha Urbaczek, Nadine Schneider, Christin Schärfer, Mathias von Beeren, Thomas Otto, and Eva Vennmann. Thank you all. I also like to thank Jörn Adomeit and Christian Rhein for the technical support and Melanie Geringhoff and Janna Eich for their office work.

At the end, I like to thank my family, Andrea, Klaus and Katharina for the support during my time at the university and Carolin for her patience, dedication and especially her view of the world.

Contents

1 Introduction

The search for new drugs constitutes one of the most challenging problems of the 21st century. The enormous financial investments and the time span required for drug discovery campaigns reflect their complexity and the high rate of failure [1]. One of the key reasons for this is the complexity of biochemical processes. Drugs are usually small molecules that interact with disease-associated macromolecules in the organism. The prediction of such interactions itself entails a high level of uncertainty. Adding the plethora of influences that the human organism exerts on the drug molecule on its way to the interaction site renders simplistic approaches useless.

A detailed understanding of the interaction between substances and proteins is fundamental for the drug discovery process. In the year 1894, Fischer postulated the central principle for molecular interaction. His *Lock-and-Key* concept [2] states that an enzyme and its substrate are complementary in their physico-chemical and steric properties. Based on that concept, P. Ehrlich developed the notion of a drug selectively binding to a receptor [3] and Langley postulated the idea that such an interaction can lead to a molecular activation or suppression of the receptor's function [4]. The Lock-and-Key principle was extended by the introduction of Koshland's *induced fit* model [5] which describes the change in protein structure upon binding to a ligand. In accordance, the conformational selection paradigm describes the ligand selecting a protein conformation that is compatible with binding and shifts the protein configuration towards this state [6–8]. These and other advances in biology, chemistry and pharmacology have lead to the introduction of *rational drug design* in which drugs are no longer discovered in a trial-and-error process but from systematic exploration of available data sources on the basis of well-founded biological models.

The evolution in drug design and the advances in computer hardware have led to the development of computational methods to assist the experimental drug discovery process. Computational and experimental methods are complementary in nature. Computational methods allow the processing and management of large data sources and supply a mechanism for large scale predictions of molecular processes. In contrast, experimental processes are more exact in their results and therefore used to validate the virtual predictions in detail. Nowadays, computational methods are established in pharmaceutical research and are mainly used during the early stages of drug discovery.

1.1 The drug discovery pipeline

Modern drug discovery is a complex multi-stage process that can be divided into the gathering of biological information, clinical testing and the final admission to the market followed by long-term studies. The following section introduces the beginning of the drug discovery pipeline which consists of the identification of macromolecular targets, the generation of biological active

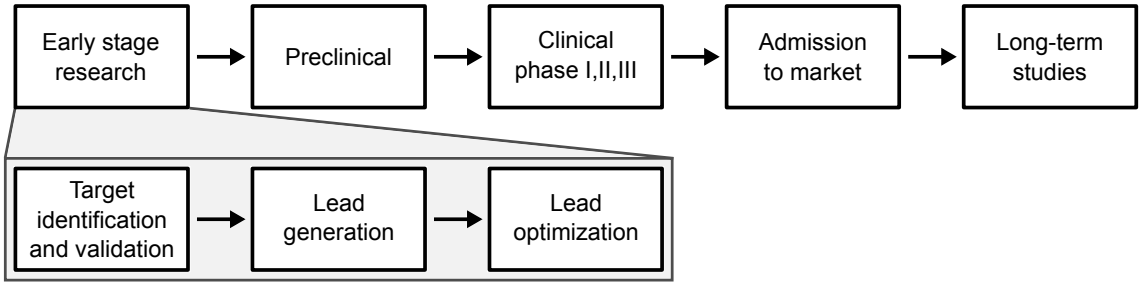


Figure 1: The drug discovery pipeline. In early-stage research phase a protein target is identified and validated. In addition, lead structures are generated and optimized. Preclinical studies assess the general safety of the potential drug in animal models and clinical phases are conducted on human probands. In clinical phase I, safety, dosage and side effects are evaluated. Studies performed in clinical phase II test the effectiveness of the drug. In clinical phase III, effectiveness is confirmed, side effects are monitored and the drug is compared to known treatments. After an admission to the market, long term studies are conducted to evaluate treatment risk, benefits and optimal usage.

structures and their optimization to preclinical drugs. In this stage, *in-silico* and *in-vitro* technologies are applied side-by-side. Figure 1 shows an overview of the drug discovery pipeline.

- **Target identification and validation**
  The first step is dedicated to the identification of a target protein associated with the disease. When such a target is found and validated, the determination of the protein structure, the substrates and interactions in the organism are beneficial for the following lead generation process. Knowledge about the protein structure and substrates allow for the prediction of the protein druggability and its interactions in the biological system. Finally, the protein is either confirmed or rejected as a suitable target.

- **Lead generation**
  After the identification of a target protein, the generation of lead structures begins. A *lead structure* is a molecule that interacts with the target protein and allows structural modifications without losing its activity. In screening campaigns, large sets of substances are experimentally tested for their biological activity. This process is supported by *in-silico* methods, e.g., for the selection of suitable screening compounds or the prediction of preferred binding modes in order to identify key interactions. Generated lead structures are validated in further screening experiments to assess their biological activity and reduce possible side effects.

- **Lead optimization** The optimization of lead structures improves their binding affinity and selectivity as well as their absorption, distribution, metabolism, excretion, and toxicology (ADMET) properties. In order to achieve such a change in molecular properties, lead compounds are structurally modified. The result of this stage is a set of potential drugs for the following preclinical studies.

1.2 Screening in drug discovery

Screening is an experimental technology to identify lead structures for a target protein. The idea is to increase the probability of finding an active compound by testing a large number of substances. This requires special hardware that guarantees a fully automated testing of thousands of compounds in a short time. These compound libraries are acquired by collecting previously tested substances in databases or through *combinatorial chemistry* that allows the generation of large chemical libraries with simple chemical reactions that describe the alternative linkage of reagents to products. The advances in automated hardware technologies and the available compound libraries led to a systematic testing of millions of compounds, the *high throughput screening* (HTS). Even though HTS is an established procedure in drug discovery, a number of problems are known. The large amount of collected data and the low signal-to-noise ratio make results difficult to interpret. In the end, only a limited number of compounds, usually below a million, can be tested in an HTS run. Estimations of the chemical space range from 10^{18} to 10^{200} with a general agreement at 10^{60} molecules [9–11]. Therefore, the selection of compounds is critical.

Virtual screening (VS) is the application of computational models to select compounds by evaluating their desirability as a lead structure [12]. It permits to evaluate millions of compounds with respect to a target protein in a short period of time at almost no costs. In general, VS is applied during the lead generation phase to select suitable compound libraries for experimental testing or to directly screen compounds. In addition, VS is used to optimize generated lead structures. The utilized technologies can be split into structure-based and ligand-based methods.

Structure-based virtual screening is applied when the protein structure and preferably the binding site of a substrate or ligand are known. Advances in X-ray crystallography and nuclear magnetic resonance (NMR) technologies have lead to reliable protein structure determination techniques that often allow for a co-identification of the protein ligand's structure [13]. The key challenge of structure-based methods is the prediction of interactions between a ligand and a protein. This *molecular docking problem* [14, 15] can be separated into three sub-problems: the generation of ligand conformations to account for ligand flexibility, the prediction of a protein-ligand complex with regard to the protein flexibility and the estimation of the corresponding binding affinity.

Ligand-based virtual screening is the search for potentially active compounds based on known ligands when no structural information about the target protein is available. If only information about the substrate or other active compounds is available, ligand-based methods scan the chemical space for similar molecules. The comparison of molecules based on their physico-chemical and steric properties lead to molecules with similar functional groups and shape. Therefore, they are expected to show similar biological activity. One way to compare molecules is to superimpose their structures. Since a superposition calculation is computationally demanding, the method is only applicable to screen small sets of compounds. In order to obtain a scalable comparison, molecular descriptors were established. A *molecular*

descriptor [16–18] captures the molecular physico-chemical and steric profile and allows a comparison without the need for a superposition. *Substructure search* is a method between a two dimensional superpositioning and a molecular similarity search. The hypothesis is that molecules that share a molecular core with known ligands have similar biological activities [19]. Two other similarity techniques are *pharmacophor modeling* [20] which obtains the key interactions between ligand and protein from known active molecules and *quantitative structure activity/property relation* (QSAR/QSPR) calculations [21] that correlate the biological activity/property of a compound with the experimentally measured biological activities.

1.3 Research goal and organization

The aim of this thesis is to present a novel algorithm named *SubSubSearch* that searches fragment spaces (FSs) for *chemical patterns* that describe molecular substructures. A fragment space describes a possibly large chemical space with molecular fragments and connection rules that specify the alternative linkage of fragments to products. Today, almost every major software used in drug discovery supports the search for patterns in molecules. Several decades of research regarding the corresponding mathematical problem [22, 23] have led to a number of subgraph isomorphism algorithms [24–34]. Unfortunately, little effort was spent on evaluating the different algorithms regarding their applicability to molecular data or to further develop those methods to work on fragment spaces. The first publication of this thesis [A1] evaluates two commonly used subgraph isomorphism algorithms on a large set of chemical patterns and molecules. On the basis of the obtained results, a subgraph isomorphism algorithm was chosen as basic component of *SubSubSearch*. The second publication of this thesis [A2] describes and evaluates the basic functionality of searching fragment spaces for molecules including a user-defined chemical pattern. The chemical pattern language used in *SubSubSearch* allows a generic description of patterns including the specification of the chemical surrounding of an atom, a feature commonly used by chemists. The third publication of this thesis [A3] explains the algorithmic advances needed to process such atomic environments. In addition, two cooperations are presented that led to the development of a visualization concept for chemical patterns [A4] and the analysis of conformational spaces regarding small molecules [A5].

The application field of *SubSubSearch* in the drug discovery pipeline lies in the lead generation and optimization phase. When structural knowledge about the substrate or known active compounds is accessible, the method allows to search combinatorial chemical spaces for new lead structures. Possible positions and variations for lead optimization are annotated on found compounds. Therefore, the method can be used to generate possible structural modifications during the lead optimization stage. The search in fragment spaces is especially attractive since the covered chemical space often includes substances that are not present in common compound databases.

In the following course of this thesis, Section 2 introduces the preliminaries on graph theory, molecular representation, chemical pattern language and chemical fragment spaces.

Section 3 reviews the algorithms for subgraph isomorphism and their use in molecular database systems. This section also includes an overview of the concept of Markush structures and their representation in database systems. Section 4 presents the published work and describes a method to optimize fragment spaces regarding the physico-chemical properties of their covered molecules. Section 5 summarizes the work of this thesis and provides an outlook of the future work.

2 Preliminaries

The following section supplies the graph theoretical background and its application to molecular data. In addition, chemical pattern languages and the concept of fragment spaces are introduced. Readers with a strong background in cheminformatics are recommended to continue with the literature review of chemical pattern search methods and applications in Section 3.

2.1 Graph theoretical background

A *graph* $G = (V, E)$ is a pair of *nodes* V and *edges* E in which each edge $e \in E$ connects two adjacent nodes $v_1, v_2 \in V$. A graph is *connected* if every node is reachable by a path of adjacent nodes from every other node. A *labeled graph* holds arbitrary labels for the nodes and/or edges. A graph is *simple* if the edges are unweighted and undirected. In an *unweighted*, *undirected* graph, the edges are uniformly weighted and have no orientation between their adjacent nodes. A *general* graph is not restricted in the graph's structure. In the following, all graphs are connected, labeled, simple and general, except when stated otherwise.

Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are *isomorphic* if a one-to-one mapping between their nodes V_1 and V_2 exists and nodes $v_1, v_2 \in V_1$ are connected if and only if their images $w_1, w_2 \in V_2$ are connected. If graphs G_1 and G_2 are labeled, the label of mapped nodes $v_1 \in V_1$ and edges $(v_1, v_2) \in E_1$ must agree on an arbitrary compatibility criteria to the labels of the corresponding images $w_1 \in V_2$ and $(w_1, w_2) \in E_2$, respectively. An *induced subgraph* of $G = (V, E)$ is a graph $G' = (V', E')$ in which the nodes $V' \subset V$ and edges $E' \subset E$ are a subset of the original graph. An edge $e = (v_1, v_2) \in E$ connecting two nodes $v_1, v_2 \in V$ are in E' if and only if $v_1, v_2 \in V'$. An *induced subgraph isomorphism* between a query graph G_1 and a target graph G_2 exists if G_1 is isomorphic to an induced subgraph of G_2, i.e., G_1 is contained in G_2.

The problem of detecting a subgraph isomorphism between two general graphs is known to be NP-complete [22, 23]. Yet, the problem on restricted graphs, e.g., planar graphs, can be computed in polynomial time [35, 36].

A closely related problem is the detection of a *maximal common subgraph* (MCS) between two graphs. A *common subgraph* of two graphs G_1 and G_2 is a graph G_{12} that is isomorphic to a subgraph of G_1 and a subgraph of G_2. Hence, the MCS of two graphs is the largest of the common subgraphs. The term MCS is mostly used to refer to the *Maximal Common Induced Subgraph* (MCIS) which is the largest common subgraph with regard to the number of common nodes. Closely related to the MCIS is the Maximal Common Edge Subgraph (MCES). A MCES is the largest common subgraph with respect to the number of edges both graphs have in common. The differences between MCIS and MCES, the algorithms that solve the problem and the applications in drug discovery are reviewed in [A6]. Even though the concept of MCS detection between molecular structures was reviewed at the beginning of my research, the development of a search routine in fragment spaces was focused on the detection of subgraph isomorphisms.

2.2 Molecular representation and comparison

A *molecule* is a group of atoms held together by covalent chemical bonds. Modeling molecules as graphs results in a *molecular graph* in which nodes represent atoms and edges donate bonds. A molecular graph is labeled to account for atom and bond properties. The *degree*, that is the number of edges attached to a node, is limited by the number of bonds an atom can form. The number of edges in a molecule linearly depends on the number of atoms. Therefore, a molecular graph is a graph of bounded degree. In order to account for the spatial arrangement of atoms, additional information can be annotated to a molecular graph. However, the graphs considered in the following only refer to the two dimensional molecular topology.

Two molecules are equal if and only if a one-to-one mapping of their atoms and bonds exists, i.e., the two molecular graphs are isomorphic. In order to map two nodes or edges, they must be identically labeled. If no unique mapping exists, a molecule can be a *substructure* of another molecule. In that case, a subgraph isomorphism between the two molecules exists.

In a graph-based comparison of molecules, some problems arise. *Mesomeric* structures have different bond localizations, e.g, aromatic systems. Therefore, their molecular graphs are not isomorphic, even though they represent the same molecules. In *stereoisomeric* structures, additional information regarding the relative arrangement of bonds around an atom must be annotated and compared to differentiate between them. Furthermore, molecules exist in different *tautomeric* forms that result from the migration of hydrogen atoms accomplished by a switch of adjacent single and double bonds. Mesomeric and tautomeric structures enforce a standardized construction of the molecular graphs for a correct comparison.

2.3 Chemical pattern languages

A *chemical pattern* generically defines a substructure of a molecule. A *chemical pattern language* defines a pattern using a formal language like SMiles Arbitrary Target Specification (SMARTS) [37], Molecular Query Language (MQL) [38], or Sybyl Line Notation (SLN) [39]. These languages specify a pattern in a textual line notation similar to the chemical formula of a molecule. The textual string defines the topology of the pattern and the properties of atoms and bonds, e.g., element symbol, charge or bond order. In addition, the SMARTS language allows an alternative logical description of atoms and bonds and the specification of an *atomic environment* that defines the chemical surrounding of an atom. The interpretation of a pattern description results in a *pattern graph* in which the nodes and edges hold a generic description of atoms and bonds, respectively. In SMARTS, an atomic environment is a node property that is modeled as an addition pattern graph. In the following, the SMARTS language is used to define chemical patterns and their visualizations are generated using the SMARTSviewer [A4].

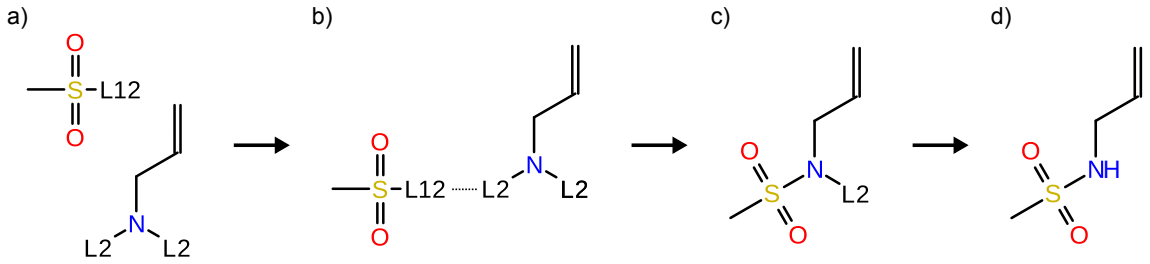


Figure 2: The construction of a product from a fragment space. a) A methyl-sulfone and a propanamine fragment. b) The connection rules allow the construction of sulfonamides by joining linkers $L2$ and $L12$. c) A sulfonamide product allowing the attachment of further fragments at linker $L2$. d) A sulfonamide molecule obtained by saturating $L2$ with hydrogen.

2.4 Chemical fragment spaces

A *fragment space* (FS) is a compact, combinatorial description of a possibly large chemical space. It consists of *fragments* that are molecules with open valences and a set of rules defining their alternative linkage to *products*. The concept of fragment spaces was introduced by Lewell et al. [40] with the REtro-synthetical Combination Analysis Procedure (RECAP). The RECAP describes chemical motifs that are easily formed by combinatorial chemistry. These rules define how molecules can be retro-synthetically cleaved to obtain fragments and, at the same time, how fragments can be combined to form products. The combinatorial nature of FSs allows the construction of novel products that were not present prior to the retro-synthetical cleavage. Figure 2 shows an example of the construction of an sulfonamide product.

Fragment spaces can be classified according to their source and application. *Generic* FSs are designed to cover the diversity of the chemical universe suitable for drug discovery [41, 42]. They are obtained from retro-synthetical cleavage of molecular libraries. Their coverage of novel molecules has shown to be a valuable source of information during lead generation and optimization [43–47]. Often, these molecules must be further modified to be synthetically accessible or new synthesis routes need to be established. In *combinatorial* FS, the fragments and connection rules directly follow chemical synthesis steps [48, 49]. Usually, a central core fragment can be decorated with different terminal groups to obtain products. A combinatorial FS is preferably used during lead generation and optimization. The application of generic and combinatorial FSs has shown that an unspecific coverage of chemical space can lead to the generation of molecules with undesired physico-chemical properties that are not suited as lead structures [50]. A *focused* FS tries to circumvent these problems [51–53]. It is constructed by a retro-synthetical cleavage of known bioactive molecules. The cleavage rules are designed to conserve bioactivity. Products obtained from such a space are likely to show comparable bioactivity with higher selectivity and optimized ADMET properties. A focused FS is a valuable source of novel molecules in lead optimization.

In the following, the developed FS methods are tested and validated on generic FSs, e.g., Breaking of Retro-synthetically Interesting Chemical Substructures (BRICS) [42]. The results

obtained from a search utilizing *SubSubSearch* are used in [A2] and [A3] to generate focused FSs.

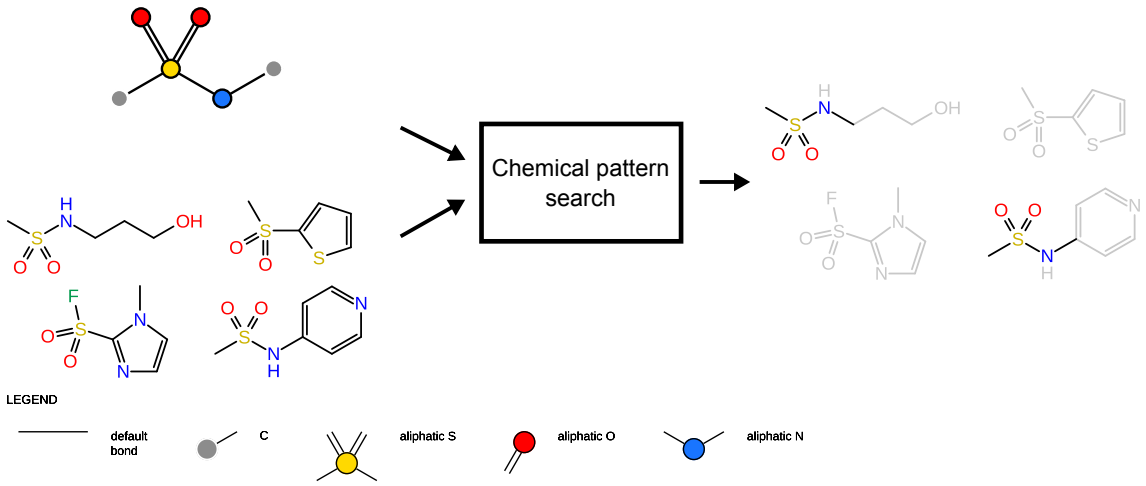


Figure 3: Chemical pattern search example. A pattern describing sulfonamides (left-top) is searched in a set of molecules (left-bottom). Sulfonamide pattern matches (right).

3 Chemical pattern search

This section gives an introduction to the algorithmic concepts of searching for chemical patterns in molecules and their application in molecular database systems. After a review of the search algorithms the concept of Markush structures which is a molecular description similar to fragment spaces is introduced. This introduction includes an overview of the development of Markush database systems and the algorithms to search these databases.

3.1 Chemical pattern search algorithms

The identification of patterns in molecules is used in virtually every field of cheminformatics. The applications include the filtering of compound sets in High Throughput Screening (HTS) [54], the detection of functional [55] or reactive groups [56] in molecules, the identification of unspecific binding molecules in protein-protein interaction assays [57], the prediction of molecular ADMET properties [58–60], and the profiling of biological activity [61]. Merlot et al. [62] and Villar et al. [63] give a detailed review of the applications for substructure search in drug discovery. Most algorithms detecting chemical patterns in molecules work on a graph representation of patterns and molecules. Therefore, the problem of searching for patterns can be solved by detecting subgraph isomorphisms between a pattern graph and a molecule graph. The following section summarizes the exact subgraph isomorphism algorithms with regard to chemical pattern recognition. Since the subgraph isomorphism problem is NP-complete, a number of heuristic approaches approximate the solution [64–68] and retrieve many molecules that include a substructure similar to the query pattern. Since *SubSubSearch* is designed to find an exact solution, approximative algorithms are not further considered in this section. Attias and Dubois [69], Barnard [70] and Willett [71] reviewed the algorithmic concept and the field of application for pattern searching in molecules and molecular databases. In the following,

the terms nodes and edges are used for chemical patterns and atoms and bonds to describe molecules.

3.1.1 Backtracking

Backtracking is a strategy applied to detect a subgraph isomorphism between a query pattern and a target molecule. It is used to iteratively generate a one-to-one mapping between a query Q and a target T. The starting point is an arbitrary node-atom pair (q, t) with $q \in Q$ and $t \in T$ which are compatible according to an arbitrary criteria, e.g., equal element symbols. A backtracking procedure subsequently extends a current partial assignment M by adding compatible node-atom pairs that are adjacent to nodes and atoms in M. During this extension, the procedure assures the same connectivity between nodes from Q and atoms from T, i.e., nodes that are connected in Q must be assigned to atoms that are also connected in T. If all query nodes are assigned to the target structure, a subgraph isomorphism between Q and T is found. If, at any point, no extension for M can be found, the procedure back-tracks to return to a previous partial assignment. Thereby, the last extension (q, t) is discarded and the process continues with an alternative assignment (q, t'). If all alternatives have been explored, the algorithm backtracks again. If the backtracking reaches the starting atom pair, no alternative starting positions are left and no subgraph isomorphism was found, the algorithm stops and the query pattern is reported to be not present in the molecule.

The first and most simple backtracking algorithm to calculate subgraph isomorphisms was described in 1957 by Ray and Kirsch [24]. Although it is extremely slow, it was the fundamental basis for modern pattern search systems that allow chemists to access millions of molecules stored in databases in retrieval times of a few seconds. Early advancements have been made by Jun et al. [28], Dengler et al. [29] and Xu [30]. They examined the number of connections present at query nodes and target atoms. If query node q has a higher degree than target atom t, the extension (q, t) is not explored because t has an insufficient number of neighbors to map all adjacent nodes of q. This leads to a pruning of the search tree and reduces the number of search paths that need to be explored. The most recent advancement in the field of backtracking-based subgraph isomorphism algorithms was made by Cordella et al. [31, 32, 34] in 2004. Their algorithm named VF and the optimized version VF2 are designed to work on directed graphs and employ a number of pruning strategies during the search. Publication [A1] describes a modified VF2 algorithm for undirected graphs including the pruning techniques in detail.

Furthermore, additional optimization can be achieved by more informative node or atom labeling, e.g., including information about an atom's neighborhood, and by rearranging the order in which pattern nodes are processed. For instance, starting or continuing with an unusual heteroatom with many bonds reduces the number of possible starting positions or increases the chances for an early backtracking, respectively. In both cases, the number of overall partial assignments is reduced and consequently the search time decreases [A1]. A clear advantage of

procedures employing backtracking strategies is the direct exploration of the structure topology. In case a small structure is assigned to a large one, the algorithm does not need to examine target atoms that are topologically too far away from the arbitrary start atom.

3.1.2 Partition and relaxation

An addition to backtracking procedures for determining the presence of a subgraph isomorphism is the technique of *partition and relaxation*. The nodes of a pattern and atoms of a molecule are arbitrarily partitioned into potentially compatible subsets and then the subsets are iteratively refined. The purpose of partitioning is the reduction of the number of possible mappings which have to be investigated during the search. In each iteration, a partition and relaxation algorithm reduces the possible assignments of query nodes to target atoms until a one-to-one mapping is established. If the algorithm encounters a stage in which at least one query node has no possible mapping left, it backtracks to the last stage. If backtracking is impossible it stops and no subgraph isomorphism between the two graphs exists. The refinement step uses a *relaxation* technique in which the node and atom descriptions are modified to include information about their direct neighbors. Thereby, information regarding further distant neighbors is iteratively collected in each node and atom description. A prominent example of such a strategy is the Morgan algorithm [72] for unique labeling of a molecular structure. Uniquely labeled molecules are easily compared for equality since the one-to-one correspondence is directly present by their atom labels.

In 1965, Sussenguth [25] presented the first partitioning algorithm for subgraph isomorphism and many followed during the 1960s and 1970s [73–76]. Sussenguth's algorithm encodes the bond type information as property of the attached atoms. This can lead to the false identification of isomorphisms. In 1972, Figueras [26] presented an advancement of Sussenguth's partition procedure that explicitly addresses bond types. Nodes and atoms of the query and target are characterized by a coding scheme retrieved from atom properties. For each query node, the algorithm generates two characteristic bit vectors: One from the comparison of node descriptions to possible mappings of target atoms. The other one by inspecting query nodes and target atom neighborhoods with respect to the edge and bond types. Both vectors are combined by logical multiplication to reduce the set of possible assignments. If an empty vector is encountered, no subgraph isomorphism is present. Also, if the set of query nodes is larger than the set of target atoms, the algorithm stops, since, at that point, a one-to-one mapping is impossible. The method uses 'higher order' connectivity to avoid false identification of isomorphic structures. The information about further distant nodes is included in the generation of the second characteristic set. In rare cases, the algorithm still finds false isomorphisms. A backtracking step would suffice to detect those cases but the author consciously chose to avoid the backtracking in order to reduce the search time and point out that false identifications only occur in rare cases [26].

Even though Ullmann [27] published his depth-first matrix-based partition algorithm in 1976, it is still one of the most extensively used subgraph isomorphism algorithms for molecular structure comparison [77, 78]. Studies by Willet et al. conducted prior to the development of the VF2 algorithm suggest that it is the best suited algorithm for chemical structures [70]. The Ullman algorithm works on the adjacency matrices of the query pattern and target molecule. In addition, it uses a $n \times m$ compatibility matrix M of bit values where n is the number of query nodes and m the number of target atoms. A non-zero entry at $(i, j) \in M$ indicates the correspondence of query node i with target atom j according to an arbitrary compatibility criteria. The procedure processes the matrix top-down. In each iteration, it choses a non-zero entry (i_1, j_1) in a row and sets all other row entries (i_1, x) with $x = (1, 2, 3, ..., n), x \neq j_1$ to zero. A refinement step relaxes the rest of the matrix by iteratively removing all mappings that became invalid. The algorithm inspects the immediate neighborhood of a possible correspondence between node x and atom y to determine if the mapping (x, y) became impossible by the assignment (i_1, j_1). For each node x' adjacent to x, a correspondence to an atom y' adjacent to atom y must exist, i.e, the matrix must have a non-zero entry at (x', y'). If not, the mapping (x, y) is invalid and the corresponding matrix entry is set to zero. If the algorithm encounters a row of all zero entries indicating a query atom that has no corresponding target atom to be assigned to, the algorithm backtracks and explores an alternative assignment. If no alternative exists, the algorithm backtracks again. If no backtracking is possible, the algorithm stops and no subgraph isomorphism exists. If the process reaches the nth row, a matrix is produced with only one non-zero entry in every row and column. Therefore, a subgraph isomorphism is detected and the one-to-one correspondence is given by the current matrix. The Ullman algorithm is well suited for internal parallelization of the partition step and external data parallelization when more than one target structure is searched. Wiepke and Rogers [79] first explored the parallelization of pattern search. Willet et al. [80] studied an internal and external parallelization of the Ullman algorithm on a AMT Distributed Array Processor (DAP). A DAP is a single-instruction-multiple-data-architecture. Experiments showed a small overall superiority of the internal over the external approach with regards to advantages for either of the two approaches depending on the input data. It was suggested to use a mixed approach for an overall good performance. Other hardware solution include an Ullmann algorithm implementation on field programmable gate arrays (FPGAs) [81–85]. A FPGA is an integrated circuit that can be custom-configured using a hardware description language. The program description is loaded prior to program execution to 'wire' the FPGA. Experiments indicated that the search speed is dominated by the program transfer overhead up to a query size of 16 atoms. For larger queries, the FPGA approach is faster than a comparable desktop computer.

In general, partition and relaxation algorithms are particularly well suited for internal parallelization over the partitions and external parallelization when more than one target structure is processed. An additional advantage is the ability to process nodes and atoms in an arbitrary order. This advantage is at the same time a major drawback since in the case of a small query and a large target, e.g., searching a functional group in a protein, many target

atoms have to be examined that are topologically too far apart. Nevertheless, optimization strategies exploring the rearrangement of nodes and atoms are not restricted in any way. A detailed evaluation of the rearrangement and an external parallelization is given in [A1].

The above backtracking and partition and relaxation algorithms solve the problem of assigning one query pattern to one target structure. Both must be explicit in their description and topology. However, a generic node description, leaving the topology explicit, can be processed with appropriate compatibility criteria allowing the assignment of generic nodes to explicit atoms, e.g., a generic query node describing all halogens is allowed to be assigned to a chlorine atom.

SubSubSearch is based on a backtracking procedure, since a one-to-one subgraph isomorphism step is utilized during the search for chemical patterns in fragment spaces. To be more specific, *SubSubSearch* uses a modified VF2 algorithm [A1] to detect parts of a pattern in fragments of a fragment space.

3.2 Molecule database systems

The number of chemical structures either purchasable or synthetically accessible has been constantly growing over the past decades [86, 87]. These structures must be organized such that an easy and fast retrieval of molecules with specific properties is possible. Database technologies offer such an advanced organization of molecular data even though the structural representations are manifold and a standard format is missing [88]. Detailed reviews regarding molecular databases are given by Stobaugh [89], Wagener [90], Hicks [91, 92], Bardor and Lardy, [93], Barnard [70], Paris [94] and Miller [95]. The following gives a general overview of the main concepts and reviews the most recent developments as well as some historical and quite successful systems.

A database usually stores explicit chemical structures in the form of connection tables and annotated additional information, e.g., molecular properties, coordinates, solvent, counter ions, or synthesis protocols. The underlying mechanisms to search a database are far from trivial. The most simple search query from a users perspective is an exact match, in which the search probes the database for the presence of a fully specified structure. A closely related scenario is a substructure or pattern search in that a query explicitly or generically specifies a part of a structure, e.g., a functional group, and the system retrieves molecules that contain the query. Today's molecular databases provide many different queries such as similarity searches, physico-chemical range searches or chemical name searches. Nevertheless, probing a database for the presence of a substructure or chemical pattern is one of the most challenging tasks since it usually requires a subgraph isomorphism calculation between the query structure and database molecules. Since the graph isomorphism comparison is a time consuming process, most database systems perform a screening step [96–99] prior to the actual isomorphism calculation. Screening is a technique to quickly identify structures that can not match the query. Most

screening steps use molecular descriptors [18] that encode features or fragments of the chemical structures and allow a fast comparison, e.g., via bit-operations. Often, the database entries are pre-processed such that the full screening information for each entry is annotated and thereby the screening performance is enhanced.

In the 1970s, two systems emerged that use a hierarchical tree structure for fragment-based screening. The Chemical Information System (CIS) developed at the National Institute of Health (NIH) [100,101] supports an atom-centered fragment as well as a ring perception screen. The fragment search processes a hierarchical tree description starting at a central atom and evaluating directly adjacent neighbor atoms with respect to the atom types and bond orders. Once all fragments are processed, the method retrieves the database structures incorporating the atom centered fragments. As a second screening, CIS searches for the presence of ring structures based on ring properties such as a ring hash code, ring size, atom types, hetero atom position and ring substitution pattern. The actual substructure search is a simple backtracking procedure to determine an atom-by-atom correspondence between the query structure and database molecule. The system does not allow generic queries and requests a manual invocation of the two screening steps prior to the substructure search. CIS also offers to search the database for molecular properties, e.g., molecular weight, atom and ring count and frequency. In 1984, CIS was stopped as a government service and it is unclear to what extend the service is still offered as part of commercial products.

The Description, Acquisition, Retrieval and Correlation (DARC) system [102–105] uses the Fragments Reduced to an Environment which is Limited (FRELs) descriptor that describes two concentric atom layers around a central atom. DARC stores the database molecule in a hierarchical tree structure with a generalized form called fuzzy FRELs at each level of the tree. Fuzzy FRELs incorporate generic bond partners and are annotated to allow a fast comparison to fuzzy FRELs generated from a query. The use of fuzzy FRELs is essential since the query pattern only specifies a part of the structure in which not all bonding partners are known. During the search, the FREL screen is followed by a bit screen match focused at ring systems and a backtracking atom-by-atom search. The DARC system supports the specification of generic queries by allowing the specification of alternative atom and bond types, number of adjacent atoms, atoms in rings, and disconnected structures. Today, the DARC system, at least the Markush version, is offered by Questel as a commercial product.

The same year in which CIS was transfered to the private sector, von Scholley [106] introduced a system using reduced graphs and hyper structures to represent database molecules. In *reduced graphs*, multiple atoms are collapsed to single nodes resulting in smaller graphs. Nodes may represent cyclic or acyclic pieces of the molecule or repetitive groups of carbon and heteroatoms. A method utilizing reduced graphs must account for the loss of information as a result of the reduction. *Hyper structures* represent more than one molecule in a single structure and are constructed by joining structures based on their common parts. Often, the origin of hyper structures is unknown after their construction and therefore hyper structures may include 'ghost' structures that were not present in the original sources. Both, reduced graphs

and hyper structures, are less demanding on the storage requirements and due to their reduced description allow a faster search. Generic queries must be formulated in a system-specific language that directly describes reduced graphs since the language offers such constructs as 'C(n)' describing a chain of n carbon atoms. The search, preceded by a limited-environment fragment bit string screen, is a partition and relaxation backtracking procedure without an atom-by-atom assignment. Since an explicit subgraph isomorphism step is missing and the information problems in reduced graphs and the ghost structures in hyper structures are not explicitly addressed, the method generates false-positive isomorphism matches [70]. Therefore, it can only be used as a screening step.

The S4 system [91, 92] was developed in 1990 by Softron GmbH in cooperation with the Beilstein institute and is used in-house by Beilstein and in the DIALOG online system [107] provided by ProQuest. Unfortunately, not much information about the detailed function of the system is available. As can be surely obtained from Hicks' system evaluation [92], S4 stores all permutation of a connection table for a database molecule in one hierarchical tree. Most likely, they account for a common labeling between query and database molecules and accomplish a structure search by traversing the tree. Substructure matches are presumably retrieved by a partial tree traversal starting inside the tree and returning results when the complete query was found, though not necessarily reaching the leafs of the trees. Storing all permutation of a connection table is similar to the approach by Messmer and Bunke described later in this section.

Another hyper structure model was developed at the University of Sheffield by Brown et al. [108–110]. In a database pre-processing step, the molecules are superimposed to generate hyper structures which are stored in an hierarchical tree structure. The superpositioning of molecules is based on the Maximal Overlapping Set which is a technique similar to the Maximal Common Subgraph. A screening step using a subset of the Chemical Abstract Service (CAS) [111] structure dictionary obtains structures from the query and compares them to dictionary structures present in hyper structures. The screening procedure accounts for ghost structures and therefore detects structures not present prior to the hyper structure construction. Hyper structures passing the screen are submitted to a parallel subgraph isomorphism calculation based on the Ullmann algorithm.

In 1993, Christine et al. [112] presented a system using database long bit strings for a screening step. Predefined structural features are encoded by bit strings in which each bit represents the presence of a feature in a database molecule. The method retrieves all relevant structural features from a query and screens the database by combining all corresponding feature bit strings by logical multiplication. The resulting bit string holds a non-zero entry for all molecules that pass the screen. An atom-by-atom search verifies the presence of the query in each remaining molecule. Unfortunately, the subgraph isomorphism algorithm used for the atom-by-atom assignment is not specified in detail and experiments showing the capabilities of the method are missing.

In the late 1990s, Messmer and Bunke [113–115] presented a direct lookup method based an a decision tree approach. The system allows a polynomial search time when discarding the time needed to construct the database. Database molecules are represented as connectivity matrices and preprocessed such that all permutations of a connectivity matrix are obtained. The database stores permuted matrices in one decision tree structure for all molecules. Queries are found by traversing the tree without the need for a backtracking step. If a tree leaf is reached, the query structure is present in the database. Substructure queries are performed by traversing the tree until the complete substructure is processed, not necessarily starting at the root or reaching a leaf. Even though such a traversal is possible in polynomial time, the decision tree itself and its construction is exponential in size and, of course, the listing of all present substructures is also exponential. The authors report different strategies for pruning the search tree that either restrict the search to graph isomorphism in which only structures equal to the query can be detected or will lead to an exponential search time, though reducing the memory demands to a fraction.

In 1997, a substructure search system [116] based on circular fragment codes emerged that supposably circumvents the need for an atom-by-atom search. Database molecules are encoded by their atom-centered circular fragments. These fragments encode the atomic environments present in circular layers around a center atom. A search request compares the fragments present in a query with those fragments present in database molecules. The system reports a match if a molecule contains the exact set of circular fragments present in the query. Since a substructure only encodes a part of a molecule, a circular fragment obtained from a substructure might miss bonding partners that are present in corresponding molecules. Therefore, a direct comparison of substructure and molecule fragment codes is problematic. The authors describe that missing substructure parts are indicated by dummy atoms but it remains unclear how these dummy atoms are generated during the computation of fragment codes. In addition, a description of the comparison between generic and explicit fragment codes is missing. The authors state that there is no prove that the correct subgraph isomorphism is obtained and if an atom-by-atom search for further verification is necessary.

OrChem [117], presented in 2009, is an open-source chemistry front-end for Oracle databases. In addition to other functionalities, OrChem allows to search molecules for the presence of substructures in a two-step procedure. Fingerprint screening uses a modified subset of PubChem fingerprints [118] and is accomplished by a parallel subgraph isomorphism search using the VF2 algorithm. The VF2 search was enhanced by employing a primary sorting step based on the frequency of atoms present in the database and a secondary sorting step ordering the atoms by their bond degree. Therefore, the VF2 processes unusual atoms first, which reduces the search space and consequently the search time.

At the same time, Golvin and Hendrick [119] developed a database design which allows a substructure search to be conducted as a single SQL query. The system stores molecules in a relational database with a database scheme directly encoding atoms and bonds. An SQL query of the form "SELECT ... FROM ... WHERE ..." is suited to select molecules with

compatibility constraints specified in the *WHERE* clause. A substructure search is a two-step procedure: the method generates an SQL query from an input substructure and retrieves the molecules performing the query. The SQL generation is accomplished by building a spanning tree over the query and filling the *FROM* and *WHERE* clauses by traversing the tree from the root to the leaves.

WebCSD [120], presented in 2010, is an online portal to the Cambridge Structural Database (CSD) [121] of the Cambridge Crystallographic Data Center (CCDC). The portal supports two-dimensional queries on the three-dimensional structures. The system employs simple screens working on chemical information obtained from the query, e.g., elements and bond types. An early version of the system used the Ullmann algorithm for subgraph isomorphism comparison and was recently replaced by a custom breadth-first search backtracking procedure. In general, the system supports explicit substructure queries. The only generic property allowed to be specified is the size of the smallest ring that an atom or bond is contained in.

The ABCD Chemical Cartridge [122–124] was developed at Johnson & Johnson Pharmaceutical Research & Development, L.L.C., starting in 2007. The system supports fully generic pattern queries using the SMARTS language. In order to employ an effective screening step database, molecules are indexed using structural keys developed in-house that encode atom and bond properties, rings of up to eight atoms, paths with a length of up to four atoms, and clusters that are atoms with more than three bonds. Additionally, the index scheme captures combinations of atomic properties which frequently occur in SMARTS queries, e.g., '[cH]' represents an aromatic carbon with exactly one hydrogen attached. An enhanced version of the Ullmann algorithm performs the atom-by-atom verification between SMARTS queries and database molecules. Optimizations include the ordering of query atoms based on their frequency in the database and the optional restriction to retrieve only one mapping opposed to all possible atom-by-atom mappings.

AMBIT-SMARTS [125] is a recent development and represents and extension of the open source Chemical Development Kit (CDK) [126]. The CDK supports the SMARTS language with various extensions. A SMARTS pattern search follows the conventional two step procedure of a screening step followed by an atom-by-atom verification. The screening phase uses a dictionary of circular-fragment keys that are pretested to occur in approximately 50% of the database molecule. The threshold is motivated from the fact that structural keys included in too many molecules result in a poor screening performance and keys that are too rare are also rare in queries. A second screen utilizes hashed keys (Daylight fingerprint approach [37]) that encode paths of up to 7 atoms. Since SMARTS queries can include generic atom and bond descriptions which interfere with the calculation of explicit keys and hash codes, the method trims all generic parts and calculates the screening information from the resulting query description. In addition, the method compares 'skeleton' keys retrieved from a query to keys precalculated from molecules. A skeleton key only captures topological information by discarding all atom and bond descriptions. The system performs a final verification on molecules passing the screening using the backtracking algorithm of Ray and Kirsch [24].

Today, database systems are one of the major components in storing and organizing large data sources for biological, chemical and pharmaceutical research. They all use concepts of preprocessing, screening and searching to allow chemists to browse through millions of molecules with response times in the order of seconds. These techniques are designed to process explicit molecules and require major modifications to be applied to combinatorial descriptions of chemical spaces such as fragment spaces. In *SubSubSearch*, a search for molecules including user-defined chemical patterns was realized to scan alternative molecule storage concepts such as fragment spaces.

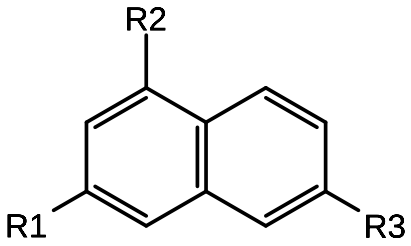


Figure 4: An example of a Markush structure. *R*-groups indicate variable attachment points. At *R*1, any halogen atom can be attached. A variable chain of three to five carbons or an aryl-group which is any aromatic ring can be attached at *R*2. Valid substituents for *R*3 are OH, any thioalkyl and 2-pyridyl.

3.3 Markush database systems

Markush structures define a series of homologous molecules by a molecular core that describes the essential central structure with attached R-groups for which series of alternative substituents are specified. Figure 4 shows an example. Markush structures are used in combinatorial chemistry, for the description of QSAR/QSPR analysis and to protect intellectual property in the form of chemical patent claims. The techniques to store, retrieve, search and compare Markush structures are the same in all three areas of application. In contrast, the Markush descriptions differ substantially. In combinatorial chemistry and QSAR/QSPR analyses the Markush structures are well defined and mostly follow general conventions. On the other hand, a patent text offers a compact and generic description by a mix of textual and graphical elements. In general, a drawn structure depicts one or more molecular cores. A textual and graphical description further characterizes the substituents. A patent text usually includes a definition of variations on substituents, indicates multiple positions for their attachment, describes repetitions of structural parts, and formulates generic expressions to represent a class of substituents. Additionally, the text covers logical relations or restrictions between attachment points, i.e., the substitution at one point enforces or reduces the possible substitutions at another point. Unfortunately, the patent literature language is not standardized and patent claims show a broad variation in terminology, grammar and style. Therefore, the concepts of Markush database systems to store chemical patent information are the most advanced in the application fields. Even though the following is focused on patent information, the described concepts are also applicable when handling Markush structures in combinatorial chemistry or QSAR/QSPR analysis.

The early Markush systems used fragment codes for indexing and searching a database. These codes are small groups of connected atoms and bonds that characterize the different chemical structures. The lack of overall structural information and the problem of different structures sharing the same fragment codes resulted in an overall poor precision of retrieval. Topological systems emerged that cover the structural relation by explicitly modeling atoms and connecting bonds. Today, Markush databases often use reduced graphs to describe the covered molecules. A reduced graph stores single atoms, explicit structures or generic descriptions in its

nodes and incorporates the overall topological information in its edges. The concept of fragment codes is still used as a screening step prior to the actual search. In the following, three major Markush database systems for chemical patent information are introduced.

Downs and Barnard [127] recently reviewed chemical information systems including the automated text analysis of chemical patents. Simmons [128] gives a historical overview of Markush systems. More details on the problems of Markush structures and the difficulties encountered in searching a Markush database for structural components are discussed by Leland et al. [129] and Welford et al. [130], respectively.

The GENeric Structure LAnguage (GENSAL) project [131–145] conducted from 1979 to 1995 at Sheffield University was concerned with the substantial problems of representing, storing and searching the information present in chemical patents. Reviews of the project are given by Lynch and Holliday [146], Downs and Barnard [147] and Bishop et al. [148]. The GENSAL pattern language [149] formalizes the textual, visual and descriptive information typically found in patents. A GENSAL query is converted into a logic tree structure [150] that describes the relation between common and variable parts as present in Markush structures. The database holds Markush information in a reduced graph description and annotates the alternative connections between these components. In the system, a reduced graph holds cyclic structures in single nodes and, therefore, allows Markush components to be represented as trees. Neither the query nor the database description allow repetitions of structural parts even though they are present in most patents. A three stage search system first uses a dictionary-based fingerprint screen, in which fingerprint fragments can span over multiple Markush components, followed by a reduced graph screen that matches graph nodes to query nodes. The second step is considered a screen since reduced graph nodes might have more than one correspondence. A one-to-many correspondence occurs for cyclic parts because their reduced graph representation is a single node with additional parameters, e.g., a six-membered ring containing five carbons and one hetero atom. The final refined search based on an adapted Ullman algorithm resolves ambiguities and accomplishes an atom-by-atom or group-by-group assignment.

MARPAT [151,152] was developed by the Chemical Abstract Service (CAS) and launched in 1988. The system is still offered online by STN international. In MARPAT, a Markush structure description is modeled as a multiple connectivity node (SnMCN) representation which is a graph that connects all Markush components. The SnMCN representation includes a logical description of connectivity conditions at multiple-used attachment points that includes the constraints textually described in a patent. From the SnMCN representation a generic MCN (GnMCN) representation is derived and both are overlayed to form a composite MCN (CpMCN) representation that allows a switch between explicit and generic description of Markush components. The GnMCN description incorporates a predefined hierarchical description of abstraction for generic nodes, e.g., a pyridine is labeled as 6-membered N-heterocycle which itself is a heterocycle and so on. The order allows a shift between the levels of abstraction which is essential for the assignment between generic components. In addition, generic nodes are attributed with group properties, e.g., charge, element counts, ring size, which would provide

a more discriminant matching but is reported to be discarded during the search [153,154]. Even though Fisanick [152] explicitly states that repetitions of structural parts variations are handled by a comparison of range attributes stored in GnMCNs, it seems that at that time, the system supported the storage of attributes, representing a base for a following attribute comparison but that comparison was never implemented as part of the search. In a search, the system generates a CpMCN for the query which is compared to the CpMCN of the database. In order to reduce the retrieval time, a screening step is conducted based on explicit fragments that are generated from SnMCNs. A backtracking procedure accomplishes an atom-by-atom or group-by-group assignment. The assignment procedure compares (more) explicit and generic nodes on the abstraction level of the more generic structure. For that final comparison, the system allows to define a Match LEvel (MLE) to control the assignment between generic query nodes (GnMCNs) and explicit (SnMCNs) or generic (GnMCNs) Markush components. If the MLE is set to 'class' as opposed to 'atom', explicit query nodes are 'changed' into their more generic representation and compared to generic nodes in the database. The feature allows the retrieval of approximate matches to the query.

Markush DARC [155, 156] is a commercial system based on the DARC software originally developed by Dubious [103]. Part of the DARC project was a collaboration with Telesystems to develop Generic DARC that allowed generic queries on explicit database structures. A further development of Generic DARC conducted by Telesystems, Derwent Publications Ltd. and the French Patent Office (INPI) resulted in Markush DARC, a system to place generic queries on Markush databases and the Merged Markush Service (MMS) now offered by Thomson Reuters, which is a unification of the Markush Pharmasearch [157] database started in 1986 by INPI and the WPI Markush database developed by Derwent Publications Ltd.. Markush DARC stores the Markush components as reduced graphs in which nodes can hold complete substructures or generic group expressions (or homologous series), e.g., CHK identifies an alkyl. The system requires a graphic query specification that allows the same degree of abstraction as present in the Markush components, i.e., allows to define super nodes that specify substructures or generic groups. The input is converted to an internal graph representation and matched against the database as present in the DARC system. The fundamental screening step in DARC is based on FRELs and reused as generic FRELs [104] in Markush DARC. A bit screen based on molecular attributes, e.g, element symbol, bonds or ring systems, complete the screening procedure which is followed by an atom-by-atom assignment. For such an assignment generic parts are translated based on a structure dictionary to explicit structures and matched atom-by-atom. The early version of the system was limited to only a subset of generic expression used in patents and lacked transparency on the translation between generic and explicit nodes. Therefore, the user had either to deal with missing matches, e.g., ethyl would not match an alkyl, or had to manually define the translation between nodes. The majority of these limitation were later resolved [156] to allow full transparency on the node translation process. Today, Markush DARC is still offered by Questel and is part of the MMS database.

The three systems were reviewed and compared from an end-user perspective [158, 159] and with regard to the database content [160] as well as technical concepts [153]. The most recent publications are a detailed comparison between Markush DARC and MARPAT [154] and a historic overview concerning all three systems [161].

Many chemical software packages allow the management of Markush structures using databases. Domine and Merlot [162] give an overview of the different systems. ChemAxon in cooperation with Thomson Reuters supports Markush database searches based on a screening step using two-dimensional structural keys followed by an atom-by-atom matching of the query against all enumerated products that pass the screen. MDL Central Library allows the storage of Markush description and offers a Markush space search that is a conventional molecule search on the full set of enumerated products. The RS3 database system by Accelrys stores and searches Markush structures and returns a result as a generic description of virtual matches and, unfortunately, does not offer the enumeration of complete products. Daylight supports virtual libraries in their Monomer Toolkit. Libraries are described in the Daylight CHUCKLES and CHORTLES notations and generic searches are formulated in the CHARTS language. A non-enumerated search is sometimes impossible, e.g., when the query is a branching structure and the stored structures are made of monomers. In that case, the search procedure works on enumerated products.

Markush database systems, regardless of the field of application, allow to search generic combinatorial descriptions of chemical spaces. They are directed at handling the generic information present in Markush structures. In fragment spaces, structural building-blocks are explicit and constrained repetitive descriptions are not allowed. Therefore, the algorithmic concepts, especially the central concept of comparing reduced graphs with varying degrees of abstraction, can not directly be transfered to fragment spaces. In addition, the search facilities provided by the different database systems seem to not support an explicit substructure or chemical pattern query. At least, the found molecules are not guaranteed to include the exact query.

3.4 Substructure search in fragment spaces

Over the last 15 years, very little academic attention was spent on searching fragment spaces. In 2007, Domine and Merlot submitted the Patent Application US 2007/0260583 A1 [162] for fast substructure searching in non-enumerated chemical libraries. The method searches an explicit query substructure in a combinatorial chemical space description similar to a fragment space. In order to retrieve combinations of fragments including a user-defined substructure, the method uses a modified Ullmann algorithm that allows many-to-one correspondences between query substructure nodes and target fragment atoms. The algorithm assigns the query to each fragment such that fragment link atoms are assigned to multiple query nodes. Therefore, the query is partially assigned to a fragment with missing query parts assigned to fragment linkers. The assignment to fragment linkers indicates that the missing substructure part must be part of a fragment that can be connected at the used link atom. After partially assigning the query to each fragment, the method explores link compatibilities to construct combinations of fragments that lead to products including the query substructure. Therefore, the algorithm finds substructures that span over multiple fragments. Unfortunately, the patent lacks a detailed description of the modified Ullmann search and the product reconstruction. According to personal communication, the method was only used as part of an in-house system at Serono.

4 Method

In this section, the published work on pattern search algorithms and fragment spaces is summarized. First, an empirical study is presented that characterizes the algorithmic behavior of two subgraph isomorphism algorithms when applied to search for chemical patterns in molecular data. As a conclusion, one algorithm was incorporated in *SubSubSearch* as basic pattern search component. Further on, the section describes the search strategy employed by *SubSubSearch* to detect recursively defined chemical patterns in fragment spaces. In addition to the published work, the section includes a method to automatically optimize fragment spaces to exclude the formation of chemical patterns in the encoded products. The methods developed to realize a search in fragment spaces were also applied in the fields of structure visualization and molecular space analysis. The end of the section covers cooperations that led to a visualization concept for SMARTS patterns and an analysis of the conformational space of small molecules.

4.1 Chemical pattern search in molecules

A first step towards the development of a pattern search method for fragment spaces was to obtain a deep understanding of subgraph isomorphism algorithms and their behavior when applied to molecular data. In drug development, such algorithms are frequently applied to browse molecular databases. These databases store millions of compounds [86, 87] and allow chemists to retrieve molecules that obey predefined structural and physico-chemical properties. One of the central requirements of these databases is the retrieval of compounds with user-defined functional groups or molecular cores. Provided that query pattern and target molecules are donated as graphs, a comparison can be realized via subgraph isomorphism techniques. Ullman introduced one of the oldest, yet frequently applied algorithm. A number of other methods followed with the most recent being the VF2 algorithm. These algorithms are applicable on general graphs and, therefore, do not pose restrictions on the graph structure in any way. Some comparisons between these algorithms have been conducted on general graphs of medium to large size [33, 34, 163, 164]. However, molecular graphs are small and by no means general. They usually comprise less than 100 atoms and are restricted in their degree by the linear number of bonds an atom can form. In addition, nodes and edges are always labeled to describe atom and bond properties which must be addressed during graph comparison. Therefore, the conducted tests give little insights into the algorithms' behavior when applied to molecular data. One of the main reasons why such a comparison was not performed in the past was the lack of suitable benchmark data sets. The focus of my work was, therefore, set on the introduction of various such benchmark sets and the discussion of the differences between the Ullmann and the VF2 subgraph isomorphism algorithm.

The major difference between the Ullmann and the VF2 algorithm is the way they process the topology of the pattern. The Ulmann algorithm constructs a compatibility matrix which represents all possible mappings of pattern nodes to molecule atoms. It processes the matrix

top-down and in every step fixes one node-atom mapping and checks all other entries for validity. Therefore, it processes pattern nodes in an arbitrary, non-topological order. The VF2 algorithm starts at an arbitrary node-atom assignment and iteratively adds node-atom pairs until all pattern nodes have a correspondence in the molecule. Therefore, it directly explores the topology of the pattern and molecule.

The benchmark sets are designed to obtain insights into the algorithms' behavior with regard to the pattern and molecule size, the type of the pattern, the order in which pattern nodes are processed, and the symmetry often present in molecules. The sets comprise various SMARTS patterns collected from literature [42, 54–56, 165–169] and molecules selected from the public ZINC database [86].

The results show that both algorithms are applicable to molecular data with an average pair-matching time below 1 millisecond. In direct comparison, the topological exploration of the VF2 algorithm is orders of magnitudes faster in all cases and more robust against outliers. The impact of pattern and molecule size on the runtime of both algorithms was found to be exponential and linear, respectively. This result is in accordance with a theoretical analysis of the subgraph isomorphism problem [30, 170]. The SMARTS language allows the definition of atomic environments which require an additional subgraph isomorphism step during the actual comparison. As can be expected, the inclusion of such a definition in a pattern description leads to an exponential increase in the runtime. The Ullmann algorithm was much more sensitive to the inclusion of atomic environments. An overall analysis of data-separation-based parallelization showed good scaling behavior for both algorithms. A similar result was obtained in an evaluation of the Ullman algorithm conducted by Willett et al [80]. An interesting finding of the study was the sensitivity of the VF2 algorithm concerning the order in which it processes the nodes of the pattern. The development of a re-arrangement scheme that conserves the topology of the pattern and places 'unusual' pattern nodes first led to a reduction in runtime of up to 13-fold compared to the original order. A detailed description of the benchmark data sets and the summarized experiments can be found in [A1].

As a conclusion of this study, a modified version of the VF2 algorithm [A1] was incorporated in *SubSubSearch* for subgraph isomorphism search.

4.2 Chemical pattern search in fragment spaces

Although conventional databases cover large numbers of molecules, their storage capacity is limited. Estimations of the chemical space relevant for drug discovery average around 10^{60} molecules. Since the largest present databases cover up to a billion compounds [171], they obviously miss many relevant structures. Provided that a chemical space is represented by a fragment space, current algorithms support similarity search procedures [44,172], exploration of novel molecules by de-novo design [47,173–175] and the creation of focused libraries [48,176,177]. The central algorithmic problem any fragment space processing method must address is the

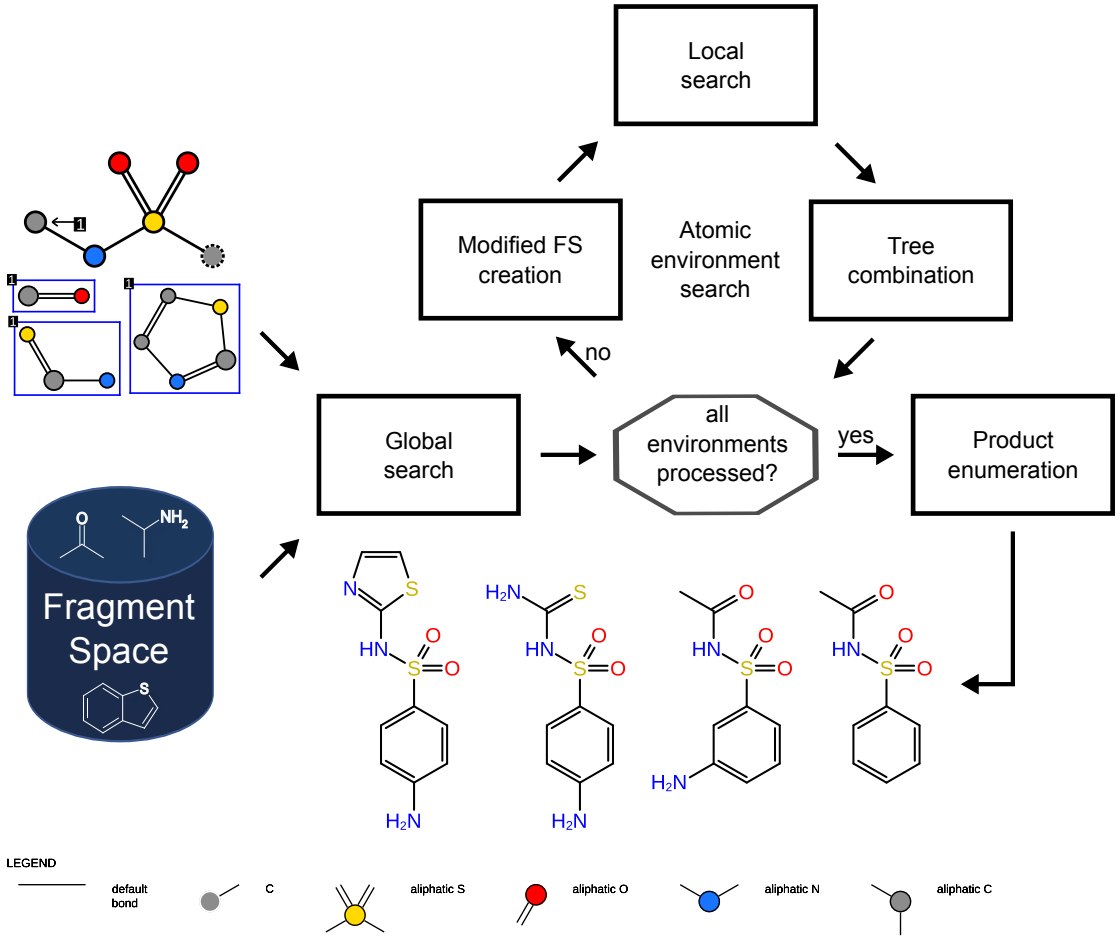


Figure 5: An example of the *SubSubSearch* processing a sulfonamide pattern with three alternative atomic environments in three main steps: the *global search* scans the fragment space for the global pattern that is the query pattern without the atomic environment information. The obtained global result is subsequently modified by the *atomic environment search* to obey the atomic environment information. From each recombination tree of the global result, the atomic environment search creates a modified fragment space. A local search scans this modified space for the presence of one atomic environment. Each obtained local recombination tree is combined with the current global recombination tree. The atomic environment search is repeated until all three environments are processed. From the final result, the *enumeration* procedure generates products containing the query pattern and the atomic environments.

combinatorial nature of such spaces. A search that only considers fragments and neglects their possible connections will almost always fail. For example, a pattern that spans over multiple fragments can only be found when the complete product space is scanned. Since a search for molecules containing a user-defined pattern is essential in drug development and a method to perform such a search in fragment spaces was missing at the time, the focus of this thesis was the development of an algorithm to search non-enumerated fragment spaces under structural constraints.

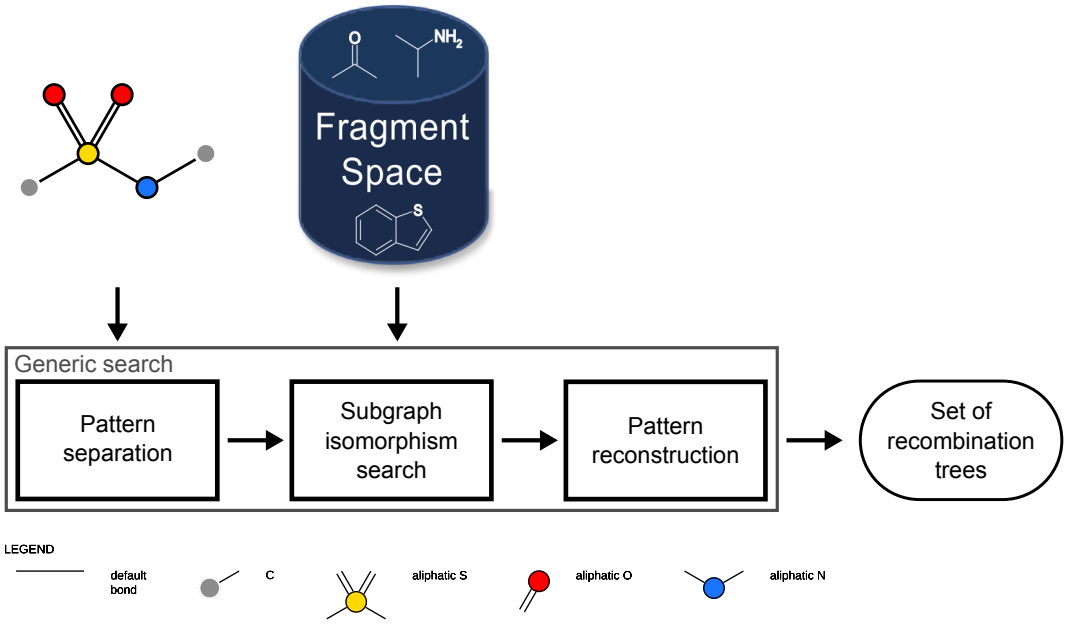


Figure 6: An example of the *generic search*. The procedure processes a sulfonamide pattern in three main steps: the search separates a pattern into all possible sub-patterns (SPs) and searches these SPs in fragments of the fragment space. The procedure reconstructs the pattern to obtain a set of recombination trees that describe sulfonamide products.

A search has to find the combination of fragments leading to products that include the query pattern. Since the SMARTS language allows the definition of recursive atomic environments, *SubSubSearch* is split in three parts: a global search, an atomic environment search and an enumeration of products as illustrated in Figure 5. Both searches use a generic search procedure.

The *generic search* shown in Figure 6 and explained in detail in [A2] is used by the global and the atomic environment search. It scans for patterns without recursive atomic environments. In order to avoid a costly enumeration during the search, the method separates the initial query into *sub-patterns* (SPs). An SP is a connected part of the initial pattern in which cyclic parts are fully contained. For that purpose, the algorithm identifies cyclic and non-cyclic parts using a bi-connected component algorithm summarized in Appendix C.1.2 and assigns cut-positions to all non-cyclic connections. Sub-patterns are generated by enumerating all possible combinations of cut-positions as described in Appendix C.1.3. Missing pieces are indicated with dummy link nodes allowing a reconnection of SPs into the initial pattern. Figure 7 shows an example of the separation process. The SPs are searched inside fragments using a modified VF2 subgraph isomorphism algorithm [A1]. During the search, dummy nodes are assigned to fragment linkers to avoid an exploration of fragment connections. The algorithm records a list of matching fragments for each SP. For the construction of products containing the complete query pattern, these lists are used as nodes to build *recombination trees*. In such a tree, two nodes are connected if and only if their SPs can be connected and the fragment link atoms are compatible with regard to the connection rules of the fragment space. The lists are split prior to the tree construction

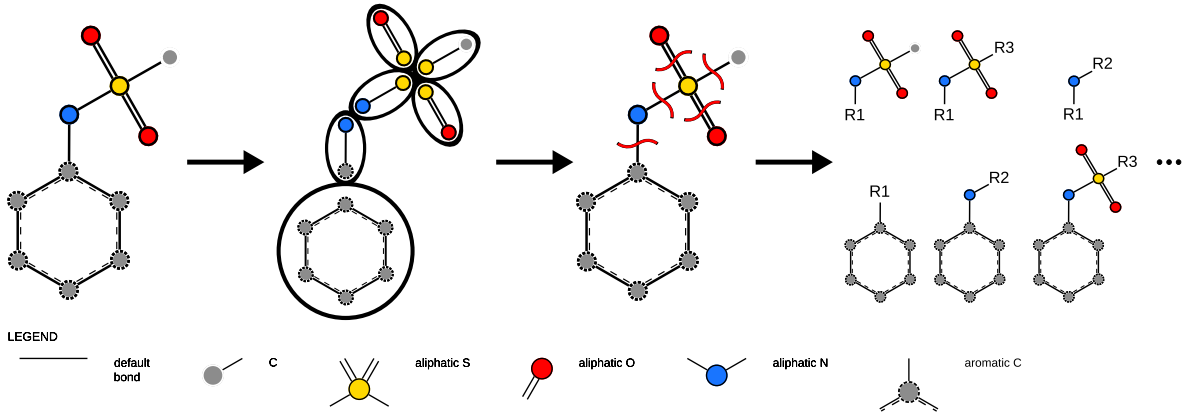


Figure 7: A sulfonamide query pattern is separated into sub-patterns (SPs). In a first step, the algorithm identifies cyclic and non-cyclic parts and assigns cut-positions to all non-cyclic connections. Sub-patterns are generated by enumerating all possible combinations of cut-positions and attaching dummy link nodes that indicate missing parts of the pattern. Note that only a subset of the enumerated SPs is shown.

such that each list only stores fragments with the same linker assigned to a dummy node. Therefore, the tree construction process must only compare the link compatibility once for every node connection. This comparison is independent of the number of fragments stored in each node and, therefore, the number of link compatibility comparisons is reduced. The resulting recombination trees describe different separations of the initial pattern and allow the enumeration of products containing this pattern of interest. Figure 8 illustrates an example of a recombination tree.

The *atomic environment search* subsequently extents recombination trees to include atomic environment definitions. An atomic environment is a pattern that defines the chemical neighborhood of an atom and can again contain atomic environments. When searching molecule databases, such an environment is either present in a molecule and can be detected, or the molecule does not contain the environment. In fragment spaces, an environment is either present in fragments itself and can be directly found, or it can be indirectly detected by attaching additional fragments, or the space does not contain products including the environment. In order to differentiate between these cases, *SubSubSearch* starts with a global search utilizing the generic search to detect combinations of fragments that include the *global pattern* which is the query pattern neglecting the atomic environment information. This global result is a collection of recombination trees. The result is iteratively refined to include the initially neglected environment information. For each modification of a tree, the atomic environment search constructs a modified fragment space. The space contains the fragments and connection rules of the initial space and is enriched with modified fragments from the global tree under consideration. For each fragment, the link atoms are renamed and connection rules are added that only allow the connection of these fragments to form products described by the global tree. Unmatched link atoms are also renamed and rules are added that prohibit connections

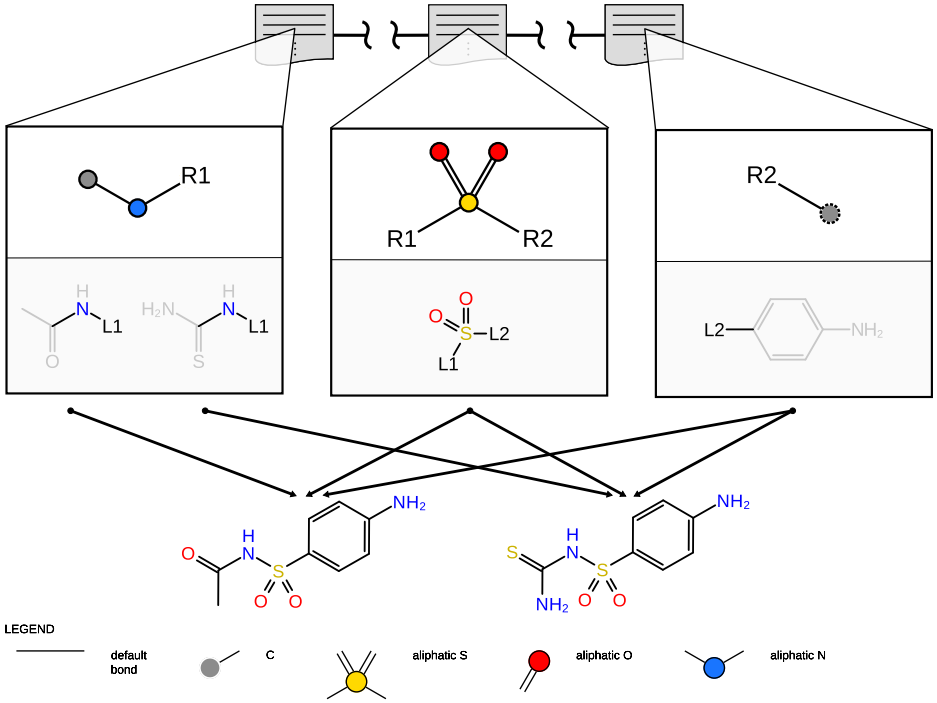


Figure 8: A recombination tree of a sulfonamide search. The sulfonamide pattern was separated into three parts shown as three nodes (top). Each node contains the sub-pattern (SPs) (box-top) and the corresponding matching fragments (box-bottom). The original pattern is reconstructed by joining the SPs at dummy link nodes $R1$ and $R2$, respectively. The enumeration procedure creates two sulfonamide products (bottom), by connecting one fragment from each node of the recombination tree at link atoms $L1$ and $L2$, respectively.

of matched fragments that do not form the global pattern. In addition, these rules allow the attachment of unmatched fragments with regard to the original link atom. Figure 9 shows an example of a modified fragment space creation.

The *local search* using the generic search scans the modified space for the presence of atomic environments. Since an environment defines the surrounding of an atom, the local search is restricted to start at the corresponding atom in fragments added from the global recombination tree. The result of the local search is a set of recombination trees describing combinations of fragments that include the atomic environment. The atomic environment is not present if the set of local recombination trees is empty. In that case, the global recombination tree is not modified. Otherwise, each of these trees either describes the environment present in a connection of fragments only using previously matched linkers, in which the environment is directly present in the global tree or the local tree connects fragments using previously unmatched linkers. In that case, the additional fragments need to be attached to the global tree to also match the atomic environment. If an atomic environment includes other atomic environments, the procedure recursively modifies the current local tree to include the additional environments before the global tree is modified. *SubSubSearch* repeats the atomic environment search with the modified global recombination tree until all atomic environments are included in the result.

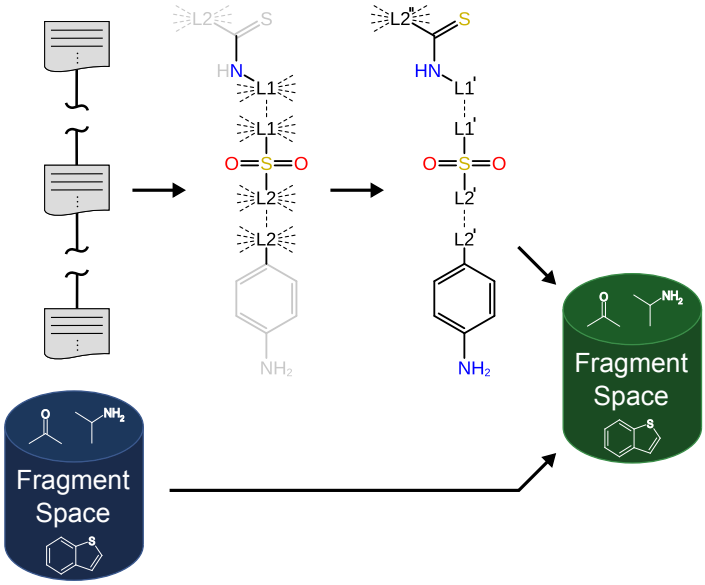


Figure 9: The procedure creates a modified fragment space from a single recombination tree obtained from a sulfonamide search. The modified space contains the complete initial space and is enriched with three fragments from the recombination tree. The matched linkers L_1 and L_2 are renamed to L_1' and L_2' and connection rules are added that only allow an assembly of these fragments to a sulfonamide product. Unmatched linker L_2 is renamed to L_2'' such that a linkage with matched fragments is prohibited and an attachment of new fragments is still allowed, e.g., if L_2 is compatible to L_1, the new linker L_2'' allows a connection to L_1 but not L_1'.

The final result of the search is a collection of recombination trees describing the assembly of fragments that incorporate the initial pattern including the complete atomic environment information. A detailed description of the procedure, especially the combination process of the global and local trees and the logical combination of atomic environments, are given in [A3].

For *enumeration* purposes as shown in Figure 8, the algorithm processes each recombination tree from the search result. For each tree, it selects one fragment from each node and connects the fragments as dictated by the tree. During this process, the same products can be created by connecting different fragments, e.g., from different recombination trees. *SubSubSearch* utilizes a database in which products are stored using a unique string representation of the product as primary keys. Therefore, equal products are neglected. A product describes a minimal set of connected fragments that include the pattern of interest. It might still contain open valences that allow the attachment of further fragments or can be saturated with hydrogens to obtain a complete molecule. In lead optimization, these attachment points can be directly explored to alter the physico-chemical properties of the obtained products.

The general applicability and usefulness of the method has been demonstrated in various use-case scenarios that mimic the search for analogues compounds, the optimization of fragment spaces to reduce their coverage of toxic substances and the extraction of large macromolecules under structural constraints. A critical factor to decide whether such a method is applicable

in a drug discovery process is the runtime required to obtain results from a search. A large scale evaluation with 738 SMARTS patterns [A1] and three fragment spaces [42, 178] revealed reasonable and robust search times. An explicit description of the experiments is given in [A2] and [A3].

Some limitations restrict the current search options. The algorithm enforces a tree-like connection of fragments into products and therefore prohibits a search for macromolecular cycles. This restriction is in agreement with most fragment space processing methods and currently available spaces. Nevertheless, in special scenarios, such a search might be interesting but would require major modification to *SubSubSearch*. Mesomeric, stereoisomeric and tautomeric structures place an exceptional challenge on any fragment space processing method. Mesomeric and tautomeric structures allow a different bond or hydrogen localization to describe the same molecule. The SMARTS language with its support for recursive atomic environments is well suited to formalize alternative mesomeric and tautomeric structures. Provided that the query definition includes at least one form present in the fragment space, *SubSubSearch* finds the desired products. The assignment of stereo centers in fragment spaces is difficult because a potential center assigned in a single fragment might vanish when fragments are connected to form symmetric products. Even if complete products are obtained by a search, they often contain open valences prohibiting an assignment of stereo centers. Therefore, a final assessment of the stereoisomeric nature can only be done on complete molecules. Since *SubSubSearch* primarily provides products, such an assignment must be part of a post-processing step.

A problem of the current implementation of fragment spaces is the handling of hydrogen atoms. As mentioned earlier, a product retrieved from a space may contain open valences that can be saturated with hydrogens to obtain a valid molecule. This termination process is currently separated from the fragments and connection rules of the fragment space, i.e., the hydrogen groups used to terminate the linkers are not in the set of fragments. Since *SubSubSearch* only processes fragments and linking rules, some products that correspond to the query after termination to molecules might be missed. This problem can be addressed by modeling the terminal hydrogen fragments as part of the fragment space. In addition, hydrogens are problematic in the SMARTS language. SMARTS allows the expression of explicit hydrogen atoms and the definition of implicit hydrogen bonding partners as a node property. When hydrogens are defined as a property, the search procedure might incorrectly identify or discard matches because the saturation of linkers with hydrogen atoms is not addressed. A solution to this problem is to enrich the fragment space with partially terminated fragments, i.e., a fragment with one linker would be present with and without an attached hydrogen.

In conclusion, *SubSubSearch* is to my knowledge the first method that allows to efficiently search fragments spaces for user-defined chemical patterns. The method finds products containing a desired pattern even if these span over multiple fragments. The search is not limited in the size of the query or in the number of fragments contained in a product. The ability to process atomic environments enable chemists to precisely define chemical patterns including local alternatives such as mesomeric and tautomeric forms. The conducted experiments show

that the procedure is able to quickly identify novel molecules which can be a valuable source in today's drug discovery processes.

4.3 Chemical pattern exclusion from fragment spaces

The application of *SubSubSearch* to optimize the properties of products contained in fragment spaces was pursued in parallel to the development of the recursive search procedure. The analyses in [A3] has shown that fragment spaces can contain products with reactive groups that are not suited for drug development process. The modification of fragment spaces to exclude such products is a tedious task when done manually. Therefore, a prototypical tool was developed [B1] that generates new fragment spaces in which formation of products containing a user-defined pattern is prohibited. A major requirement was that these fragment spaces are usable by any fragment space processing method, i.e., such a modification must be done on the basis of fragments, linkers and connection rules only.

The algorithm to exclude patterns from fragment spaces, here named *SubSubExclusion*, works in a four step procedure as depicted in Figure 10. It utilizes *SubSubSearch* to obtain a set of recombination trees that describe the products to be excluded from the space. The trees are subsequently processed in the following order: trees that consist of one node, trees that contain two nodes and trees that incorporate more than two nodes. *Fragment removal*: Trees consisting of a single node store fragments that fully include the pattern. These fragments are removed from the new space. *Exclusion over one linker*: The recombination trees that contain two nodes represent products that are composed of two fragments. The formation of these products is excluded by renaming the linkers of the corresponding fragments and generating new connection rules that prohibit the connection of these fragments, yet allow the connection to all other fragments that are compatible to the original linkers. *Exclusion over multiple linker*: If trees include more than two nodes, they describe products with more than two fragments. A direct exclusion of the connection of these fragments would lead to the loss of products that do not contain the pattern of interest, e.g., if a tree describes the connection of fragment A, B and C, the products A-B and B-C would be lost. Therefore, *SubSubExclusion* removes the fragments from the new space and adds enumerated partial products. These products allow to apply the same exclusion strategy as for products composed of two fragments. In the above example, the method enumerates the partial products A-B and B-C and excludes the connection of A-B with C and A with B-C. In general, *SubSubExclusion* generates new recombination trees in which one terminal node is removed. From these trees, all partial products are enumerated to create larger fragments. The connection of these fragments with fragments from removed terminal nodes would lead to products that include the user-defined pattern. In such a combination of fragments, the pattern spans over exactly one linker and this connection is prohibited in the new space. In order to assure that all products that contain fragments of the match and do not incorporate the pattern can be created in the new space, the algorithm enumerates partial products from the center of the recombination tree and excludes the connection to

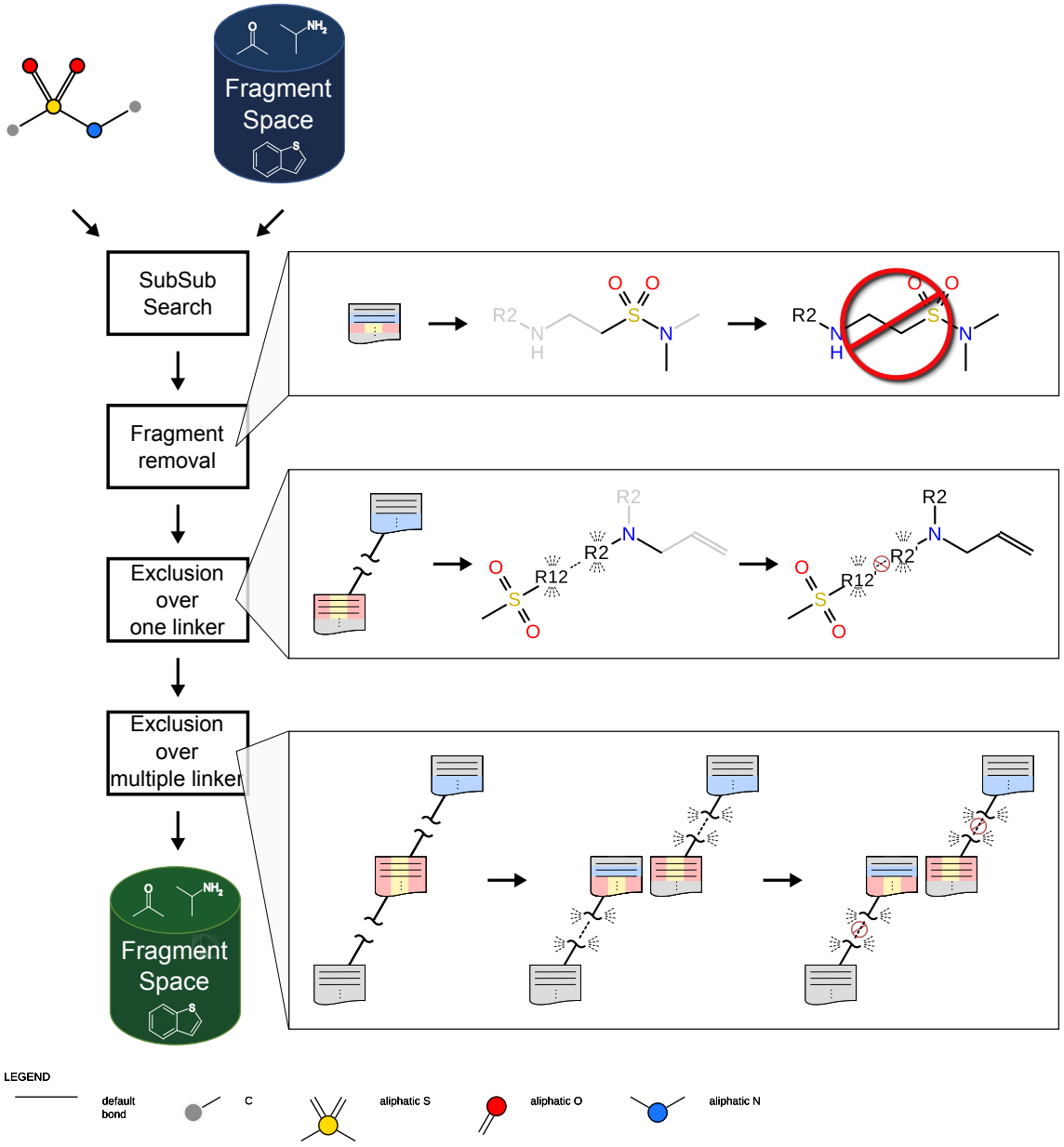


Figure 10: The *SubSubExclusion* algorithm generates a new fragment space without sulfonamide products in four steps. *SubSubSearch*: The sulfonamide pattern is searched in the input fragment space. The result is a set of three recombination trees describing sulfonamide products. *Fragment removal*: The first tree consists of a single node that describes fragments completely including the pattern. These fragments are excluded from the new space. *Exclusion over one linker*: The second tree incorporates two nodes representing a match that spans over one linker. The linkers are renamed and corresponding connection rules are generated that exclude the connection to a sulfonamide product. *Exclusion over multiple linker*: The algorithm excludes the matches over two linkers by enumerating partial products and excluding the direct connection rule between partial products. The result is a new fragment space in which the connection of fragment to sulfonamide products is prohibited.

	Smarts	Time	Matches			New fragment space		
			1 Frag.	2 Frag.	3-6 Frag.	Fragments	Linkers	Rules
easy	PAINS 1	5.52 s	1192	0	0	4799	18	66
	PAINS 4	5.82 s	56	0	0	4785	16	64
	PAINS 7	5.04 s	0	0	0	4799	16	64
	PAINS 9	6.36 s	0	904	0	4799	17	63
	PAINS 10	5.22 s	0	452	0	4799	17	64
hard	PAINS 3	9.3 h	2	1944	$2.6 * 10^8$	$4.8 * 10^6$	106	215
	PAINS 6	10.1 m	0	2202	$9.0 * 10^7$	$5.9 * 10^6$	85	334
	PAINS 8	9.3 s	3	10 530	$2.2 * 10^7$	$2.8 * 10^4$	47	161
	PAINS 11	7.2 m	0	3180	$7.0 * 10^6$	$2.2 * 10^5$	765	232
critical	PAINS 2	30.5 h	4	57 808	$4.7 * 10^9$	$3.0 * 10^6$	244	150
	PAINS 5	30.5 m	0	0	$4.9 * 10^8$	$1.0 * 10^6$	279	1418
	PAINS 12	4.2 s†	0	312	$1.3 * 10^9$‡	-	-	-

Table 1: Results for the generation of new fragment spaces excluding the formation of PAINS patterns. Shown are the runtimes in seconds (s), minutes (m) and hours (h) needed to generate new fragment spaces. The number of matches are shown with regard to the number of fragments incorporated in the individual products. New fragment spaces are characterized by the number of contained fragments, linkers and connection rules. († matching time only, enumeration time of more than 2 days. ‡ containing matches of six fragments.)

fragments in adjacent tree nodes. The center is a subtree in which all terminal nodes are removed. The method generates subtrees that describe partial products by enumerating all connected subtrees of the center. Partial products are created by enumerating the products described by the generated subtrees. For example, if a match describes a linear connection of five fragments A-B-C-D-E, the subtree B-C-D describes the center and the products B-C and C-D are enumerated. To assure that the new fragment can not be linked to form the pattern, the connection of A and D with B-C and B and E with C-D are excluded in the new space.

The method was tested in two experiments using the BRICS 4k fragment space. The BRICS space includes 4799 fragments, 16 linkers and 64 connection rules and allows the generation of 7 809 670 products with exactly two fragments and about 10^{16} constructed of up to five fragments. In the first experiment, the formation of 28 smaller patterns describing reactive groups are subsequently excluded from BRICS 4k. The time to generate a new fragment space was 32.33 s on a single Intel(R) Xeon(R) CPU E5630 @ 2.53 GHz core. The new space included 5340 fragments, 24 linkers, 71 connection rules and 7 392 254 pair-products. The rise in fragments is due to the enumeration of partial products for a few patterns that span over multiple fragments. The higher number in linkers and connection rules results from the exclusion of pair-connections. The number of pair-connections is slightly lower, which indicates that the space contains less products spanning over one linker. The new fragment space allows the application of fragment space processing methods, e.g., similarity search, and guarantees that such methods will not retrieve products that include one of the excluded patterns.

The patterns used in the first experiment are small and simple. They describe a few atoms and are mostly linear. To explore the limitations of *SubSubExclusion*, the second experiment tests 12 patterns with increased complexity including rings and multiple branches. The patterns describe compounds similar to ones that show an unspecific binding in protein-protein interaction assays [57]. From these Pan-Assay INterference compoundS (PAINS), a subset was excluded from the BRICS 4k space. At the time the experiments were conducted, *SubSubSearch* did not support patterns with an explicit hydrogen description or atomic environment specifications. Therefore, the selected PAINS were re-written by hand. Appendix C.2 shows depictions of the patterns. The results in Table 1 show that the patterns are grouped into three sets: easy, hard and critical patterns. The first section of the table shows easy patterns for which a new space can be generated in seconds. These patterns only occur in products with one or two fragments and, therefore, their formation can be directly excluded. The center section shows patterns for which the runtime increases up to hours, the number of excluded products is large and the generated spaces include a large number of fragments. The large number of products leads to an extensive enumeration of partial products which generates many new fragments and increases the runtime. These fragment spaces are difficult to process for subsequent methods due to the large number of fragments. The bottom section of the table shows the results for patterns that describe the limits of *SubSubExclusion*. For PAINS 2, the number of products is over one billion which lead to a runtime of one and a half days which was deemed too long for a single exclusion. An exclusion of PAINS 5 was accomplished in half an hour but the resulting space contained 10^6 fragments and 1418 connection rules. Such a high number of fragments and rules is most likely a problem when further processing such a space. In the last example, a new fragment space could not be generated because the runtime exceeded two days. The reason for such a long runtime is the number and complexity of found products. These products include up to six fragments leading to a high number of possible partial products. In addition, the number of slightly over one billion products places exceptional demands on the computation. This example illustrates the problem of enumerating partial products during the exclusion.

In conclusion, the presented *SubSubExclusion* prototype is a novel method that demonstrates the possibility to automatically optimize fragment spaces in regard to the structural properties of contained products. The method is applicable for small to medium patterns with low complexity. The exclusion of unspecific and complex patterns leads to long runtimes and complex fragment spaces that are critical in their number of fragments, linkers and connection rules.

4.4 Chemical pattern visualization

The communication of molecular structures via two dimensional diagrams has a long tradition in chemistry. These diagrams offer chemists an intuitive representation of molecules and a quick identification of structural features that are often essential for molecular optimization or chemical synthesis. The two dimensional representation of molecules as graphical structure

diagrams is often referred to as the 'language of chemists'. However, such an intuitive visualization of chemical patterns has never emerged. The chemical pattern languages used to describe structural parts of molecules are often cryptic line notations designed for computational use and, are therefore, hard to read for humans. Existing tools often use a hybrid visualization in which pattern features are annotated as text of atoms and bonds. These visualizations only represent a small advantage over the corresponding line representation. In order to overcome this lack of usability, we developed the SMARTSviewer, a tool that converts generic chemical patterns formulated in SMARTS language into two dimensional diagrams.

For an intuitive visualization, three components had to be realized: the development of appropriate graphical representations for SMARTS properties, a layout closely related to the form of structural diagrams and a conversion of SMARTS strings into a computer processable structure. For the visualization of pattern features, new graphical elements had to be developed. In the SMARTSviewer, atoms are depicted as colored cycles. The colors code the most common elements in organic chemistry. Atom properties are annotated at these cycles, e.g., small numbers preceded with a plus or minus sign represent the charge. A complete overview of all graphical elements to represent SMARTS language components can be found in [A4]. The SMARTSviewer also visualizes logical expressions by coding a NOT in red and an OR in blue, e.g., an atom that is not carbon is depicted as a dark gray circle surrounded by a red frame. Chemical atomic environments are shown in numbered boxes and their location is indicated with the corresponding number pointing at the described atom. Again, the color coding for logical relations is applied. The SMARTSviewer shows bonds as lines between the atoms. Each diagram is accompanied by a dynamic legend that gives a textual explanation of the different components. Figure 11 shows an example.

In order to obtain an intuitive representation, we based the overall layout on the structure diagram representation for molecules and included recommendations from the International Union of Pure and Applied Chemistry (IUPAC) for structure diagram drawing of variable structures. For an automated generation of a graphical depiction from the input of a SMARTS string, the string must be parsed and analyzed for its syntactic and semantic content. A context-free grammar [179] was developed that models the SMARTS language. Such a grammar is used in two ways: it allows the assessment of the correct syntax of a SMARTS string and it transfers the line representation into an abstract syntax tree (AST). An AST represents the semantic content in a computer processable form that can be easily processed to obtain a visual representation.

The SMARTSviewer was tested on 762 SMARTS patterns that included between 2 and 1008 characters and in 247 cases also recursive atomic environments. A diagram was successfully generated for each pattern in the set. The details are given in [A4].

In the SMARTSviewer project, my part was the engineering of the context-free grammar to convert SMARTS string into the corresponding AST representation. In addition, I contributed to the development of the concept for visualizing SMARTS language components.

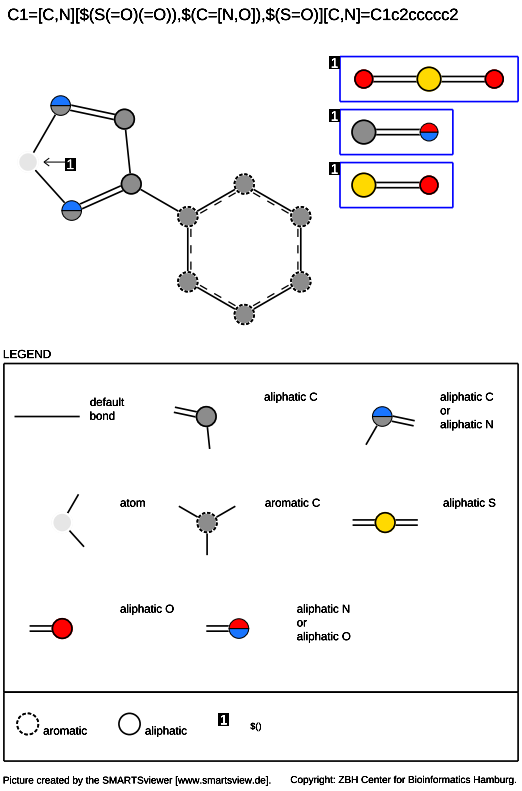


Figure 11: Visualization of a SMARTS pattern (top) created with the SMARTSviewer.

4.5 Molecular conformation analysis

Understanding the nature of small molecules is essential to create computational models for drug discovery. The conformation of a molecule is defined as the three dimensional spacial arrangement of atoms. In general, a conformation represents an energetic state of a molecule because attractive and repulsive forces act between parts of the structure. These forces rearrange a molecule into a stable conformation in which the energy is as small as possible. There may be multiple stable low-energy conformations and a molecule will undergo a transition between these states when it absorbs enough energy to pass over energy barriers in-between states. The understanding and generation of low-energy conformations are indispensable in today's drug discovery. For example, an accurate pose prediction in protein-ligand docking highly depends on the conformation of the docked ligand. Unfortunately, the knowledge about small molecule conformations only partially covers the chemical space relevant for the drug discovery processes.

We addressed this lack of knowledge with a systematical analysis of different crystal structure databases regarding the conformations observed in crystallized ligands. Our approach employs a hierarchical classification of torsion patterns. These patterns, formulated in SMARTS, describe the dihedral angle in a chain of four atoms. The hierarchy defines successively more complex and specific chemical motives to obtain a unique classification for each torsion angle. We analyzed a number of databases by assigning the torsion patterns to rotatable bonds in ligands via a pattern search algorithm. A collection of the recorded angles for each pattern is represented as

a histogram to describe their distribution. Peaks in the histograms show frequently observed torsion angles and correspond to local energy minima. The histogram obtained from analyzing the Cambridge Structure Database [121] and the Protein Data Bank [180] as well as an estimation of the covered chemical space and comparison to similar approaches are presented in [A5].

The development and adaption of the central SMARTS-based pattern search algorithm was my contribution in the development of the torsion analysis procedure. The SMARTS language was extended to describe the hybridization state of an atom and the lone pair of a nitrogen atom. The pattern search procedure was adapted to account for these changes.

5 Conclusion & Outlook

This thesis presents a novel algorithm to search for recursively defined patterns in non-enumerated fragment spaces. In a prototypical way, this algorithm was extended to automatically optimize fragment spaces regarding the physico-chemical properties of contained products. The search method allows to completely scan the chemical space covered in a fragment space description. The general applicability was achieved by placing no initial restrictions on the complexity of the user-defined pattern or the intricacy of the fragment space. The algorithm addresses the combinatorial nature of fragment spaces with a divide and conquer strategy that allows a search without the costly enumeration of products. The method has proven to be applicable in different drug discovery scenarios such as searches for analogues compounds, exploration of molecular cores, detection of toxic products, and the extraction of macromolecular structures. The obtained results are further processable in multiple ways, e.g., lead optimization, focused analysis and fragment space optimization. The experiments show that the algorithm retrieves desired products in runtimes that are considered applicable in a drug discovery pipeline. This was a major concern during the development since enumeration-based methods require a large amount of computational resources and runtime which renders them inapplicable.

At the current stage, the implementation has some deficiencies as described in Section 4.2. Stereoisomeric differences and hydrogen substitution patterns are not fully resolved. These problems can be addressed with modifications to the fragment space implementation or with additional post-processing steps. The prototypical algorithm to optimize fragment spaces has shown that such an automatic modification is possible but additional work is required to obtain an applicable tool. Especially the generation of new fragment spaces with a lower number of fragments, linkers and connection rules need to be realized.

The future work addresses these limitations and will exploit the opportunities to adapt the method to other problems. For example, the separation and reconstruction procedure to search parts of the pattern in fragments could be modified to not only detect the query as pattern in products, but to also obtain products that share a common subgraph with the query. For such a maximal common subgraph calculation, the separation phase would not only generate those sub-patterns that indicate all missing pattern parts with dummy nodes but also all possible sub-patterns in which the dummy nodes are removed. Therefore, a pattern search could terminate at any point in a fragment. The reconstruction phase would return the partial reconstruction that includes the maximal number of query nodes. Thereby, products would share a maximal common subgraph with the query. Since a maximal common subgraph comparison detects a common structural part between two graphs, the retrieved products would be structurally similar to the query. Therefore, a similarity comparison between the query and products of the fragment space would be possible.

The intermediate results supplied by *SubSubSearch* as recombination trees have proven to be valuable for the construction of a focused fragment space [A2] and the optimization of a

space regarding the physico-chemical properties of the products. The tree represents products that share a common structural feature in a very compact form. It might be suited to realize a comparison between different fragment spaces regarding the structural features contained in products. Even an extension to allow mathematical operations such as set intersection or set exclusion on fragment spaces could be possible. A set intersection would detect the common products contained in two or more fragment spaces. This would be especially useful when fragment spaces represent patent information. When two patents claim the same molecule, the younger patent is invalid which is a major concern in pharmaceutical research. In the same way would a set exclusion be useful when new patents are written.

In summary, this thesis presents novel algorithmic techniques that extent the scope of algorithms processing fragment spaces and enable chemists to access compounds that are hidden in the combinatorial nature of these spaces.

References

[1] M. Dickson and J. P. Gagnon, "The cost of new drug discovery and development," *Discovery Medicine*, vol. 4, no. 22, pp. 172–179, 2004.

[2] E. Fischer, "Einfluss der configuration auf die wirkung der enzyme," *Berichte der deutschen chemischen Gesellschaft*, vol. 27, no. 3, pp. 2985–2993, 1894.

[3] A. M. Silverstein, "Paul Ehrlich's passion: the origins of his receptor immunology," *Cellular Immunology*, vol. 194, no. 2, pp. 213–221, 1999.

[4] J. N. Langley, "On the reaction of cells and of nerve-endings to certain poisons, chiefly as regards the reaction of striated muscle to nicotine and to curari," *The Journal of Physiology*, vol. 33, no. 4-5, pp. 374–413, 1905.

[5] D. E. Koshland, "Application of a theory of enzyme specificity to protein synthesis," *Proceedings of the National Academy of Sciences*, vol. 44, no. 2, pp. 98–104, 1958.

[6] B. Ma, S. Kumar, C. J. Tsai, and R. Nussinov, "Folding funnels and binding mechanisms," *Protein Engineering Design and Selection*, vol. 12, no. 9, pp. 713–720, 1999.

[7] C. J. Tsai, S. Kumar, B. Ma, and R. Nussinov, "Folding funnels, binding funnels, and protein function," *Protein Science*, vol. 8, no. 6, pp. 1181–1190, 1999.

[8] P. Csermely, R. Palotai, and R. Nussinov, "Induced fit, conformational selection and independent dynamic segments: an extended view of binding events," *Trends in Biochemical Sciences*, vol. 35, no. 10, pp. 539–546, 2010.

[9] R. S. Bohacek, C. McMartin, and W. C. Guida, "The art and practice of structure-based drug design: A molecular modeling perspective," *Medicinal Research Reviews*, vol. 16, no. 1, pp. 3–50, 1996.

[10] T. Fink, H. Bruggesser, and J.-L. Reymond, "Virtual exploration of the small-molecule chemical universe below 160 daltons," *Angewandte Chemie International Edition*, vol. 44, no. 10, pp. 1504–1508, 2005.

[11] J.-L. Reymond, R. van Deursen, L. C. Blum, and L. Ruddigkeit, "Chemical space as a source for new drugs," *MedChemComm*, vol. 1, pp. 30–38, 2010.

[12] D. Gorse, A. Rees, M. Kaczorek, and R. Lahana, "Molecular diversity and its analysis," *Drug Discovery Today*, vol. 4, no. 6, pp. 257–264, 1999.

[13] F. Lottspeich and H. Zorbas, *Bioanalytik*. Spektrum Akademischer Verlag, Apr. 1998.

[14] M. Rarey, *Bioinformatics From Genomes to Drugs*, ch. I-7, pp. 315–360. Wiley-VCH Verlag GmbH & Co. KGaA, 2004.

[15] M. Mihasan, "What in silico molecular docking can do for the 'bench-working biologists'," *Journal of Biosciences*, vol. 37, no. 6, pp. 1089–1095, 2012.

[16] P. Willett, J. M. Barnard, and G. M. Downs, "Chemical similarity searching," *Journal of Chemical Information and Modeling*, vol. 38, no. 6, pp. 983–996, 1998.

[17] M. Rarey and J. S. Dixon, "Feature trees: a new molecular similarity measure based on tree matching," *Journal of Computer-Aided Molecular Design*, vol. 12, no. 5, pp. 471–490, 1998.

[18] R. Todeschini and V. Consonni, *Handbook of Molecular Descriptors*. Wiley-VCH, Oct. 2000.

[19] J. Klekota and F. P. Roth, "Chemical substructures that enrich for biological activity," *Bioinformatics*, vol. 24, no. 21, pp. 2518–2525, 2008.

[20] G. M. Spitzer, M. Heiss, M. Mangold, P. Markt, J. Kirchmair, G. Wolber, and K. R. Liedl, "One concept, three implementations of 3D pharmacophore-based virtual screening: distinct coverage of chemical search space," *Journal of Chemical Information and Modeling*, vol. 50, no. 7, pp. 1241–1247, 2010.

[21] J. Verma, V. M. Khedkar, and E. C. Coutinho, "3D-QSAR in drug design – a review," *Current Topics in Medicinal Chemistry*, vol. 10, no. 1, pp. 95–115, 2010.

[22] R. C. Read and D. G. Corneil, "The graph isomorphism disease," *Journal of Graph Theory*, vol. 1, no. 4, pp. 339–363, 1977.

[23] G. Gati, "Further annotated bibliography on the isomorphism disease," *Journal of Graph Theory*, vol. 3, no. 2, pp. 95–109, 1979.

[24] L. C. Ray and R. A. Kirsch, "Finding chemical records by digital computers," *Science*, vol. 126, no. 3278, pp. 814–819, 1957.

[25] E. H. Sussenguth, "A graph-theoretic algorithm for matching chemical structures," *Journal of Graph Theory*, vol. 5, no. 1, pp. 36–43, 1965.

[26] J. Figueras, "Substructure search by set reduction," *Journal of Graph Theory*, vol. 12, no. 4, pp. 237–244, 1972.

[27] J. R. Ullmann, "An algorithm for subgraph isomorphism," *Journal of the Association for Computing Machinery*, vol. 23, pp. 31–42, 1976.

[28] X. Jun and Z. Maosen, "HBA: New algorithm for structural match and applications," *Tetrahedron Computer Methodology*, vol. 2, no. 2, pp. 75–83, 1989.

[29] A. Dengler and I. Ugi, "A central atom based algorithm and computer program for substructure search," *Computers and Chemistry*, vol. 15, no. 2, pp. 103–107, 1991.

[30] J. Xu, "GMA: A generic match algorithm for structural homomorphism, isomorphism, and maximal common substructure match and its applications," *Journal of Chemical Information and Computer Sciences*, vol. 36, no. 1, pp. 25–34, 1996.

[31] L. Cordella, P. Foggia, C. Sansone, and M. Vento, "Performance evaluation of the VF graph matching algorithm," in *Image Analysis and Processing, 1999. Proceedings. International Conference on*, (IEEE Computer Society, Washington, DC, USA), pp. 1172 –1177, 1999.

[32] L. P. Cordella, P. Foggia, C. Sansone, and M. Vento, "An improved algorithm for matching large graphs," in *In: 3rd IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition, Cuen*, pp. 149–159, 2001.

[33] C. Irniger and H. Bunke, "Theoretical analysis and experimental comparison of graph matching algorithms for database filtering," in *Graph Based Representations in Pattern Recognition* (E. Hancock and M. Vento, eds.), vol. 2726 of *Lecture Notes in Computer Science*, pp. 39–52, Springer Berlin / Heidelberg, 2003. 10.1007/3-540-45028-9_11.

[34] L. P. Cordella, P. Foggia, C. Sansone, and M. Vento, "A (sub)graph isomorphism algorithm for matching large graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 10, pp. 1367–1372, 2004.

[35] D. Eppstein, "Subgraph isomorphism in planar graphs and related problems," in *Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms*, SODA '95, (Philadelphia, PA, USA), pp. 632–640, Society for Industrial and Applied Mathematics, 1995.

[36] F. Dorn, "Planar subgraph isomorphism revisited." online, 2009. arXiv:0909.4692.

[37] Daylight Chemical Information Systems, Inc. of Aliso Viejo, CA, *Daylight Theory Manual*, 4.9 ed., 2008.

[38] E. Proschak, J. K. Wegner, A. Schueller, G. Schneider, and U. Fechner, "Molecular query language (MQL) - a context-free grammar for substructure matching," *Journal of Chemical Information and Modeling*, vol. 47, no. 2, pp. 295–301, 2007.

[39] S. Ash, M. A. Cline, R. W. Homer, T. Hurst, and G. B. Smith, "SYBYL line notation (SLN): A versatile language for chemical structure representation," *Journal of Chemical Information and Computer Sciences*, vol. 37, no. 1, pp. 71–79, 1997.

[40] X. Q. Lewell, D. B. Judd, S. P. Watson, and 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 Science*, vol. 38, no. 3, pp. 511–522, 1998.

[41] P. Schneider and G. Schneider, "Collection of bioactive reference compounds for focused library design," *QSAR and Combinatorial Science*, vol. 22, no. 7, pp. 713–718, 2003.

[42] J. Degen, C. Wegscheid-Gerlach, A. Zaliani, and M. Rarey, "On the art of compiling and using 'drug-like' chemical fragment spaces," *ChemMedChem*, vol. 3, no. 10, pp. 1503–1507, 2008.

[43] G. M. Downs and J. M. Barnard, "Techniques for generating descriptive fingerprints in combinatorial libraries," *Journal of Chemical Information and Computer Sciences*, vol. 37, no. 1, pp. 59–61, 1997.

[44] M. Rarey and M. Stahl, "Similarity searching in large combinatorial chemistry spaces," *Journal of Computer-Aided Molecular Design*, vol. 15, no. 6, pp. 497–520, 2001.

[45] C. Gerlach, H. Broughton, and A. Zaliani, "Ftree query construction for virtual screening: a statistical analysis," *Journal of Computer-Aided Molecular Design*, vol. 22, no. 2, pp. 111–118, 2008.

[46] U. Lessel, B. Wellenzohn, M. Lilienthal, and H. Claussen, "Searching fragment spaces with feature trees," *Journal of Chemical Information and Modeling*, vol. 49, no. 2, pp. 270–279, 2009.

[47] T. Lippert, T. Schulz-Gasch, O. Roche, W. Guba, and M. Rarey, "De novo design by pharmacophore-based searches in fragment spaces," *Journal of Computer-Aided Molecular Design*, vol. 25, no. 10, pp. 931–945, 2011.

[48] J. Fischer, U. Lessel, and M. Rarey, "LoFT: Similarity-driven multiobjective focused library design," *Journal of Chemical Information and Modeling*, vol. 50, no. 1, pp. 1–21, 2010.

[49] U. Lessel, B. Wellenzohn, J. R. Fischer, and M. Rarey, "Design of combinatorial libraries for the exploration of virtual hits from fragment space searches with LoFT," *Journal of Chemical Information and Modeling*, vol. 52(2), no. ja, pp. 373–379, 2011.

[50] L. Weber, "Current status of virtual combinatorial library design," *QSAR and Combinatorial Science*, vol. 24, no. 7, pp. 809–823, 2005.

[51] S. C. Pegg, J. J. Haresco, and I. D. Kuntz, "A genetic algorithm for structure-based de novo design," *Journal of Computer-Aided Molecular Design*, vol. 15, no. 10, pp. 911–933, 2001.

[52] A. C. Pierce, G. Rao, and G. W. Bemis, "BREED: Generating novel inhibitors through hybridization of known ligands. application to CDK2, p38, and HIV protease," *Journal of Medicinal Chemistry*, vol. 47, no. 11, pp. 2768–2775, 2004.

[53] B. C. Pearce, D. R. Langley, J. Kang, H. Huang, and A. Kulkarni, "E-novo: an automated workflow for efficient structure-based lead optimization," *Journal of Chemical Information and Modeling*, vol. 49, no. 7, pp. 1797–1809, 2009.

[54] M. Hann, B. Hudson, X. Lewell, R. Lifely, L. Miller, and N. Ramsden, "Strategic pooling of compounds for high-throughput screening," *Journal of Chemical Information and Computer Sciences*, vol. 39, no. 5, pp. 897–902, 1999.

[55] W. Walters and M. A. Murcko, "Prediction of 'drug-likeness'," *Advanced Drug Delivery Reviews*, vol. 54, no. 3, pp. 255–271, 2002.

[56] S. J. Enoch, J. C. Madden, and M. T. D. Cronin, "Identification of mechanisms of toxic action for skin sensitisation using a SMARTS pattern based approach," *SAR and QSAR in Environmental Research*, vol. 19, no. 5-6, pp. 555–578, 2008.

[57] J. B. Baell and G. A. Holloway, "New substructure filters for removal of pan assay interference compounds (PAINS) from screening libraries and for their exclusion in bioassays," *Journal of Medicinal Chemistry*, vol. 53, no. 7, pp. 2719–2740, 2010. PMID: 20131845.

[58] A. G. Leach, H. D. Jones, D. A. Cosgrove, P. W. Kenny, L. Ruston, P. MacFaul, J. M. Wood, N. Colclough, and B. Law, "Matched molecular pairs as a guide in the optimization of pharmaceutical properties; a study of aqueous solubility, plasma protein binding and oral exposure," *Journal of Medicinal Chemistry*, vol. 49, no. 23, pp. 6672–6682, 2006.

[59] A. C. Lee, J. Y. Yu, and G. M. Crippen, "pk(a) prediction of monoprotic small molecules the SMARTS way," *Journal of Chemical Information and Modeling*, vol. 48, no. 10, pp. 2042–2053, 2008.

[60] J. Kazius, R. McGuire, and R. Bursi, "Derivation and validation of toxicophores for mutagenicity prediction," *Journal of Medicinal Chemistry*, vol. 48, no. 1, pp. 312–320, 2005.

[61] V. J. Gillet, P. Willett, and J. Bradshaw, "Identification of biological activity profiles using substructural analysis and genetic algorithms," *Journal of Chemical Information and Computer Science*, vol. 38, no. 2, pp. 165–179, 1998.

[62] C. Merlot, D. Domine, C. Cleva, and D. Church, "Chemical substructures in drug discovery," *Drug Discovery Today*, vol. 8, no. 13, pp. 594–602, 2003.

[63] H. O. Villar, M. R. Hansen, and R. Kho, "Substructural analysis in drug discovery," *Current Computer-Aided Drug Design*, vol. 3, pp. 59–67, 2007.

[64] S. Medasani, R. Krishnapuram, and Y. Choi, "Graph matching by relaxation of fuzzy assignments," *IEEE Transactions on Fuzzy Systems*, vol. 9, no. 1, pp. 173–182, 2001.

[65] J. T. Wang, K. Zhang, G. Chang, and D. Shasha, "Finding approximate patterns in undirected acyclic graphs," *Pattern Recognition*, vol. 35, no. 2, pp. 473–483, 2002.

[66] F. DePiero and D. Krout, "An algorithm using length-r paths to approximate subgraph isomorphism," *Pattern Recognition Letters*, vol. 24, pp. 33–46, 2003.

[67] E. Yu and X. Wang, "A subgraph isomorphism algorithm based on hopfield neural network," in *Advances in Neural Networks ISNN 2004* (F.-L. Yin, J. Wang, and C. Guo, eds.), vol. 3173 of *Lecture Notes in Computer Science*, pp. 436–441, Springer Berlin Heidelberg, 2004.

[68] V. Lipets, N. Vanetik, and E. Gudes, "Subsea: an efficient heuristic algorithm for subgraph isomorphism," *Data Mining and Knowledge Discovery*, vol. 19, pp. 320–350, 2009.

[69] R. Attias and J. E. Dubois, "Substructure systems: concepts and classifications," *Journal of Chemical Information and Computer Sciences*, vol. 30, no. 1, pp. 2–7, 1990.

[70] J. M. Barnard, "Substructure searching methods: Old and new," *Journal of Chemical Information and Computer Sciences*, vol. 33, no. 4, pp. 532–538, 1993.

[71] P. Willett, "Searching techniques for databases of two- and three-dimensional chemical structures," *Journal of Medicinal Chemistry*, vol. 48, no. 13, pp. 4183–4199, 2005.

[72] H. L. Morgan, "The generation of a unique machine description for chemical structures – a technique developed at chemical abstracts serviceull," *Journal of Graph Theory*, vol. 5, no. 2, pp. 107–113, 1965.

[73] R. E. Tarjan, *Graph Algorithms in Chemical Computation*, ch. 2, pp. 1–20. American Chemical Society, 1977.

[74] T.-K. Ming and S. J. Tauber, "Chemical structure and substructure search by set reduction," *Journal of Graph Theory*, vol. 11, no. 1, pp. 47–51, 1971.

[75] J. McGregor, "Relational consistency algorithms and their application in finding subgraph and graph isomorphisms," *Information Sciences*, vol. 19, no. 3, pp. 229–250, 1979.

[76] L. Kitchen and E. V. Krishnamurthy, "Fast, parallel, relaxation screening for chemical patent data-base search," *Journal of Chemical Information and Computer Sciences*, vol. 22, no. 1, pp. 44–48, 1982.

[77] A. T. Brint and P. Willett, "Algorithms for the identification of 3-dimensional maximal common substructures," *Journal of Chemical Information and Computer Sciences*, vol. 27, no. 4, pp. 152–158, 1987. s.

[78] G. M. Downs, M. F. Lynch, P. Willett, G. A. Manson, and G. A. Wilson, "Transputer implementations of chemical substructure searching algorithms," *Tetrahedron Computer Methodology*, vol. 1, no. 3, pp. 207–217, 1988.

[79] W. T. Wipke and D. Rogers, "Rapid subgraph search using parallelism," *Journal of Chemical Information and Computer Sciences*, vol. 24, no. 4, pp. 255–262, 1984. PMID: 6520146.

[80] P. Willett, T. Wilson, and S. F. Reddaway, "Atom-by-atom searching using massive parallelism. implementation of the Ullmann subgraph isomorphism algorithm on the distributed array processor," *Journal of Chemical Information and Computer Sciences*, vol. 31, no. 2, pp. 225–233, 1991.

[81] S. Ichikawa, H. Saito, L. Udorn, and K. Konishi, "Evaluation of accelerator designs for subgraph isomorphism problem," in *Proceedings of the The Roadmap to Reconfigurable Computing, 10th International Workshop on Field-Programmable Logic and Applications*, FPL '00, (London, UK), pp. 729–738, Springer-Verlag, 2000.

[82] S. Ichikawa, L. Udorn, and K. Konishi, "An FPGA-based implementation of subgraph isomorphism algorithm," *Transactions of Information Processing Society of Japan*, vol. 41, p. 3949, 2000.

[83] S. Ichikawa and S. Yamamoto, "Data dependent circuit for subgraph isomorphism problem," in *Field-Programmable Logic and Applications: Reconfigurable Computing Is Going Mainstream* (M. Glesner, P. Zipf, and M. Renovell, eds.), vol. 2438 of *Lecture Notes in Computer Science*, pp. 1068–1071, Springer Berlin Heidelberg, 2002.

[84] S. Ichikawa, H. Saito, L. Udorn, and K. Konishi, "Trade-offs in custom circuit designs for subgraph isomorphism problems," *IEICE Transactions On Information And Systems*, vol. E86D, no. 7, pp. 1250–1257, 2003.

[85] S. Yamamoto, S. Ichikawa, and H. Yamamoto, "The design and evaluation of data-dependent hardware for subgraph isomorphism problem," *IEICE Transactions On Information And Systems*, vol. E87D, no. 8, pp. 2038–2047, 2004.

[86] J. Irwin and B. Shoichet, "ZINC–a free database of commercially available compounds for virtual screening," *Journal of Chemical Information and Modeling*, vol. 45, no. 1, pp. 177–182, 2005.

[87] E. E. Bolton, Y. Wang, P. A. Thiessen, and S. H. Bryant, "PubChem: Integrated platform of small molecules and biological activities," in *Annual Reports in Computational Chemistry* (R. A. Wheeler and D. C. Spellmeyer, eds.), vol. 4 of *Annual Reports in Computational Chemistry*, pp. 217–241, Elsevier, 2008.

[88] W. A. Warr, "Representation of chemical structures," *Wiley Interdisciplinary Reviews: Computational Molecular Science*, vol. 1, no. 4, pp. 557–579, 2011.

[89] R. E. Stobaugh, "Chemical substructure searching," *Journal of Chemical Information and Computer Sciences*, vol. 25, no. 3, pp. 271–275, 1985.

[90] B. A. Wagener, "Chemical substructure searching - comparing 3 commercially available databases," *Online Review*, vol. 10, no. 3, pp. 173–183, 1986.

[91] M. G. Hicks and C. Jochum, "Substructure search systems. 1. performance comparison of the MACCS, DARC, HTSS, CAS registry MVSSS, and S4 substructure search systems," *Journal of Chemical Information and Computer Sciences*, vol. 30, no. 2, pp. 191–199, 1990.

[92] M. G. Hicks, C. Jochum, and H. Maier, "Substructure search systems for large chemical data bases," *Analytica Chimica Acta*, vol. 235, pp. 87–92, 1990.

[93] P. Bador and J. P. Lardy, "Information-systems for research on chemical structures," *New Journal of Chemistry*, vol. 14, no. 11, pp. 805–818, 1990.

[94] C. G. Paris, "Chemical structure handling by computer," *Information Today*, vol. 32, pp. 271–337, 1997.

[95] M. A. Miller, "Chemical database techniques in drug discovery," *Nature Reviews Drug Discovery*, vol. 1, no. 3, pp. 220–227, 2002.

[96] R. Nasr, R. Vernica, C. Li, and P. Baldi, "Speeding up chemical searches using the inverted index: The convergence of chemoinformatics and text search methods," *Journal of Chemical Information and Modeling*, vol. 52, no. 4, pp. 891–900, 2012.

[97] P. Liu, D. K. Agrafiotis, D. N. Rassokhin, and E. Yang, "Accelerating chemical database searching using graphics processing units," *Journal of Chemical Information and Modeling*, vol. 51, no. 8, pp. 1807–1816, 2011.

[98] R. D. Natale, A. Ferro, R. Giugno, M. Mongiov, A. Pulvirenti, and D. Shasha, "SING: subgraph search in non-homogeneous graphs," *BMC Bioinformatics*, vol. 11, p. 96, 2010.

[99] C. Solnon, "AllDifferent-based filtering for subgraph isomorphism," *Artificial Intelligence*, vol. 174, no. 12-13, pp. 850–864, 2010.

[100] R. J. Feldmann and S. R. Heller, "An application of interactive graphics-the nested retrieval of chemical structures," *Journal of Graph Theory*, vol. 12, no. 1, pp. 48–54, 1972.

[101] R. J. Feldmann, G. W. A. Milne, S. R. Heller, A. Fein, J. A. Miller, and B. Koch, "An interactive substructure search system," *Journal of Chemical Information and Computer Sciences*, vol. 17, no. 3, pp. 157–163, 1977.

[102] R. Attias, "DARC substructure search system: a new approach to chemical information," *Journal of Chemical Information and Computer Sciences*, vol. 23, no. 3, pp. 102–108, 1983.

[103] J. E. Dubois and Y. Sobel, "DARC system for documentation and artificial intelligence in chemistry," *Journal of Chemical Information and Computer Sciences*, vol. 25, no. 3, pp. 326–333, 1985.

[104] J. E. Dubois, A. Panaye, and R. Attias, "DARC system: notions of defined and generic substructures. filiation and coding of FREL substructure (SS) classes," *Journal of Chemical Information and Computer Sciences*, vol. 27, no. 2, pp. 74–82, 1987.

[105] J.-E. Dubois, "Chemical complexity and molecular topology - the DARC concepts and applications," *L'Actualite chimique*, vol. 320-21, no. 320-21, pp. 37–42, 2008.

[106] A. Von Scholley, "A relaxation algorithm for generic chemical structure screening," *Journal of Chemical Information and Computer Sciences*, vol. 24, no. 4, pp. 235–241, 1984.

[107] I. O. Hartwell and K. A. Haglund, *An Overview of DIALOG*, vol. 436, ch. 4, pp. 42–63. ACS Symposium Series, 1990.

[108] R. D. Brown, G. M. Downs, P. Willett, and A. P. F. Cook, "Hyperstructure model for chemical structure handling: generation and atom-by-atom searching of hyperstructures," *Journal of Chemical Information and Computer Sciences*, vol. 32, no. 5, pp. 522–531, 1992.

[109] R. D. Brown, G. M. Downs, G. Jones, and P. Willett, "Hyperstructure model for chemical structure handling: Techniques for substructure searching," *Journal of Chemical Information and Computer Sciences*, vol. 34, no. 1, pp. 47–53, 1994.

[110] R. D. Brown, G. Jones, P. Willett, and R. C. Glen, "Matching two-dimensional chemical graphs using genetic algorithms," *Journal of Chemical Information and Computer Sciences*, vol. 34, no. 1, pp. 63–70, 1994.

[111] J. Heyman, E. Karasinskia, and P. Giles, "CAS information services for medicinal chemists," *Drug Information Journal*, vol. 16, no. 4, pp. 185–190, 1982.

[112] B. D. Christie, B. A. Leland, and J. G. Nourse, "Structure searching in chemical databases by direct lookup methods," *Journal of Chemical Information and Computer Sciences*, vol. 33, no. 4, pp. 545–547, 1993.

[113] B. T. Messmer, *Efficient Graph Matching Algorithms for Preprocessed Model Graphs*. PhD thesis, Universität Bern, Institut für Informatik und angewandte Mathematik, 1995.

[114] B. T. Messmer and H. Bunke, "Subgraph isomorphism in polynomial time," tech. rep., Universität Bern, Institut für Informatik und angewandte Mathematik, 1995.

[115] B. T. Messmer and H. Bunke, "A decision tree approach to graph and subgraph isomorphism detection," *Pattern Recognition*, vol. 32, no. 12, pp. 1979–1998, 1999.

[116] Y. Xiao, Y. Qiao, J. Zhang, S. Lin, and W. Zhang, "A method for substructure search by atom-centered multilayer code," *Journal of Chemical Information and Computer Sciences*, vol. 37, no. 4, pp. 701–704, 1997.

[117] M. Rijnbeek and C. Steinbeck, "OrChem - an open source chemistry search engine for Oracle(R)," *Journal of Cheminformatics*, vol. 1, no. 1, p. 17, 2009.

[118] D. L. Wheeler, T. Barrett, D. A. Benson, S. H. Bryant, K. Canese, V. Chetvernin, D. M. Church, M. DiCuccio, R. Edgar, S. Federhen, L. Y. Geer, W. Helmberg, Y. Kapustin, D. L. Kenton, O. Khovayko, D. J. Lipman, T. L. Madden, D. R. Maglott, J. Ostell, K. D. Pruitt, G. D. Schuler, L. M. Schriml, E. Sequeira, S. T. Sherry, K. Sirotkin, A. Souvorov, G. Starchenko, T. O. Suzek, R. Tatusov, T. A. Tatusova, L. Wagner, and E. Yaschenko, "Database resources of the national center for biotechnology information," *Nucleic Acids Research*, vol. 34, no. Database issue, pp. D173–D180, 2006.

[119] A. Golovin and K. Henrick, "Chemical substructure search in SQL," *Journal of Chemical Information and Modeling*, vol. 49, no. 1, pp. 22–27, 2009.

[120] I. R. Thomas, I. J. Bruno, J. C. Cole, C. F. Macrae, E. Pidcock, and P. A. Wood, "WebCSD: the online portal to the cambridge structural database," *Journal of Applied Crystallography*, vol. 43, no. 2, pp. 362–366, 2010.

[121] F. H. Allen, "The cambridge structural database: a quarter of a million crystal structures and rising," *Acta Crystallographica, Section B: Structural Science*, vol. 58, no. Pt 3 Pt 1, pp. 380–388, 2002.

[122] D. K. Agrafiotis, S. Alex, H. Dai, A. Derkinderen, M. Farnum, P. Gates, S. Izrailev, E. P. Jaeger, P. Konstant, A. Leung, V. S. Lobanov, P. Marichal, D. Martin, D. N. Rassokhin, M. Shemanarev, A. Skalkin, J. Stong, T. Tabruyn, M. Vermeiren, J. Wan, X. Y. Xu, and X. Yao, "Advanced biological and chemical discovery (ABCD): Centralizing discovery knowledge in an inherently decentralized world," *Journal of Chemical Information and Modeling*, vol. 47, no. 6, pp. 1999–2014, 2007. PMID: 17973472.

[123] P. Kirkpatrick, "Informatics: The ABCD of data management," *Nature Reviews Drug Discovery*, vol. 6, no. 12, pp. 956–957, 2007.

[124] D. K. Agrafiotis, V. S. Lobanov, M. Shemanarev, D. N. Rassokhin, S. Izrailev, E. P. Jaeger, S. Alex, and M. Farnum, "Efficient substructure searching of large chemical libraries: The ABCD chemical cartridge," *Journal of Chemical Information and Modeling*, vol. 51, no. 12, pp. 3113–3130, 2011.

[125] N. Jeliazkova and N. Kochev, "AMBIT-SMARTS: Efficient searching of chemical structures and fragments," *Molecular Informatics*, vol. 30, no. 8, pp. 707–720, 2011.

[126] C. Steinbeck, Y. Han, S. Kuhn, O. Horlacher, E. Luttmann, and E. Willighagen, "The chemistry development kit (CDK): An open-source java library for chemo- and bioinformatics," *Journal of Chemical Information and Computer Sciences*, vol. 43, no. 2, pp. 493–500, 2003. PMID: 12653513.

[127] G. M. Downs and J. M. Barnard, "Chemical patent information systems," *Wiley Interdisciplinary Reviews: Computational Molecular Science*, vol. 1, no. 5, pp. 727–741, 2011.

[128] E. S. Simmons, "Markush structure searching over the years," *World Patent Information*, vol. 25, no. 3, pp. 195–202, 2003.

[129] B. A. Leland, B. D. Christie, J. G. Nourse, D. L. Grier, R. E. Carhart, T. Maffett, S. M. Welford, and D. H. Smith, "Managing the combinatorial explosion," *Journal of Chemical Information and Computer Sciences*, vol. 37, no. 1, pp. 62–70, 1997.

[130] S. M. Welford, M. F. Lynch, and J. M. Barnard, "Towards simplified access to chemical structure information in the patent literature," *Journal of Information Science*, vol. 6, no. 1, pp. 3–10, 1983.

[131] M. F. Lynch, J. M. Barnard, and S. M. Welford, "Computer storage and retrieval of generic chemical structures in patents. 1. introduction and general strategy," *Journal of Chemical Information and Computer Sciences*, vol. 21, no. 3, pp. 148–150, 1981.

[132] S. M. Welford, M. F. Lynch, and J. M. Barnard, "Computer storage and retrieval of generic chemical structures in patents. 3. chemical grammars and their role in the manipulation of chemical structures," *Journal of Chemical Information and Computer Sciences*, vol. 21, no. 3, pp. 161–168, 1981.

[133] J. M. Barnard, M. F. Lynch, and S. M. Welford, "Computer storage and retrieval of generic structures in chemical patents. 4. an extended connection table representation for

generic structures," *Journal of Chemical Information and Computer Sciences*, vol. 22, no. 3, pp. 160–164, 1982.

[134] S. M. Welford, M. F. Lynch, and J. M. Barnard, "Computer storage and retrieval of generic chemical structures in patents. 5. algorithmic generation of fragment descriptors for generic structure screening," *Journal of Chemical Information and Computer Sciences*, vol. 24, no. 2, pp. 57–66, 1984.

[135] V. J. Gillet, S. M. Welford, M. F. Lynch, P. Willett, J. M. Barnard, G. M. Downs, G. Manson, and J. Thompson, "Computer storage and retrieval of generic chemical structures in patents. 7. parallel simulation of a relaxation algorithm for chemical substructure search," *Journal of Chemical Information and Computer Sciences*, vol. 26, no. 3, pp. 118–126, 1986.

[136] V. J. Gillet, G. M. Downs, A. Ling, M. F. Lynch, P. Venkataram, J. V. Wood, and W. Dethlefsen, "Computer storage and retrieval of generic chemical structures in patents. 8. reduced chemical graphs and their applications in generic chemical structure retrieval," *Journal of Chemical Information and Computer Sciences*, vol. 27, no. 3, pp. 126–137, 1987.

[137] G. M. Downs, V. J. Gillet, J. D. Holliday, and M. F. Lynch, "Computer storage and retrieval of generic chemical structures in patents. 9. an algorithm to find the extended set of smallest rings in structurally explicit generics," *Journal of Chemical Information and Computer Sciences*, vol. 29, no. 3, pp. 207–214, 1989.

[138] G. M. Downs, V. J. Gillet, J. D. Holliday, and M. F. Lynch, "Computer storage and retrieval of generic chemical structures in patents. 10. assignment and logical bubble-up of ring screens for structurally explicit generics," *Journal of Chemical Information and Computer Sciences*, vol. 29, no. 3, pp. 215–224, 1989.

[139] W. Dethlefsen, M. F. Lynch, V. J. Gillet, G. M. Downs, J. D. Holliday, and J. M. Barnard, "Computer storage and retrieval of generic chemical structures in patents. 11. theoretical aspects of the use of structure languages in a retrieval system," *Journal of Chemical Information and Computer Sciences*, vol. 31, no. 2, pp. 233–253, 1991.

[140] W. Dethlefsen, M. F. Lynch, V. J. Gillet, G. M. Downs, J. D. Holliday, and J. M. Barnard, "Computer storage and retrieval of generic chemical structures in patents. 12. principles of search operations involving parameter lists: matching-relations, user-defined match levels, and transition from the reduced graph search to the refined search," *Journal of Chemical Information and Computer Sciences*, vol. 31, no. 2, pp. 253–260, 1991.

[141] V. J. Gillet, G. M. Downs, J. D. Holliday, M. F. Lynch, and W. Dethlefsen, "Computer storage and retrieval of generic chemical structures in patents. 13. reduced graph generation," *Journal of Chemical Information and Computer Sciences*, vol. 31, no. 2, pp. 260–270, 1991.

[142] J. D. Holliday, G. M. Downs, V. J. Gillet, and M. F. Lynch, "Computer storage and retrieval of generic chemical structures in patents. 14. fragment generation from generic structures," *Journal of Chemical Information and Computer Sciences*, vol. 32, no. 5, pp. 453–462, 1992.

[143] J. D. Holliday, G. M. Downs, V. J. Gillet, and M. F. Lynch, "Computer storage and retrieval of generic chemical structures in patents. 15. generation of topological fragment descriptors from nontopological representations of generic structure components," *Journal of Chemical Information and Computer Sciences*, vol. 33, no. 3, pp. 369–377, 1993.

[144] J. D. Holliday and M. F. Lynch, "Computer storage and retrieval of generic chemical structures in patents. 16. the refined search: An algorithm for matching components of generic chemical structures at the atom-bond level," *Journal of Chemical Information and Computer Sciences*, vol. 35, no. 1, pp. 1–7, 1995.

[145] J. D. Holliday and M. F. Lynch, "Computer storage and retrieval of generic chemical structures in patents. 17. evaluation of the refined search," *Journal of Chemical Information and Computer Sciences*, vol. 35, no. 4, pp. 659–662, 1995.

[146] M. F. Lynch and J. D. Holliday, "The sheffield generic structures projecta retrospective review," *Journal of Chemical Information and Computer Sciences*, vol. 36, no. 5, pp. 930–936, 1996.

[147] G. M. Downs and J. M. Barnard, "Chemical patents and structural information - the sheffield research in context," *Journal of Documentation*, vol. 54, pp. 106–120, 1998.

[148] N. Bishop, V. Gillet, J. Holliday, and P. Willett, "Chemoinformatics research at the university of sheffield: a history and citation analysis," *Journal of Information Science*, vol. 29, no. 4, pp. 249–267, 2003.

[149] J. M. Barnard, M. F. Lynch, and S. M. Welford, "Computer storage and retrieval of generic chemical structures in patents. 2. GENSAL, a formal language for the description of generic chemical structures," *Journal of Chemical Information and Computer Sciences*, vol. 21, no. 3, pp. 151–161, 1981.

[150] J. M. Barnard, M. F. Lynch, and S. M. Welford, "Computer storage and retrieval of generic chemical structures in patents. 6. an interpreter program for the generic structure description language GENSAL," *Journal of Chemical Information and Computer Sciences*, vol. 24, no. 2, pp. 66–71, 1984.

[151] W. Fisanick, "Storage and retrival of generic chemical structure representation." http://www.freepatentsonline.com/4642762.html, 1984.

[152] W. Fisanick, "The chemical abstract's service generic chemical (Markush) structure storage and retrieval capability. 1. basic concepts," *Journal of Chemical Information and Computer Sciences*, vol. 30, no. 2, pp. 145–154, 1990.

[153] J. M. Barnard, "A comparison of different approaches to Markush structure handling," *Journal of Chemical Information and Computer Science*, vol. 31, no. 1, pp. 64–68, 1991.

[154] A. H. Berks, "Current state of the art of Markush topological search systems," *World Patent Information*, vol. 23, no. 1, pp. 5–13, 2001.

[155] E. A. Ferns, K. E. Shenton, and M. I. Norton, PaAnd Langdon, "Development of the derwent Markush graphics database for patients," *Abstracts Of Papers Of The American Chemical Society*, vol. 192, p. 36, 1986.

[156] P. Benichou, C. Klimczak, and P. Borne, "Handling genericity in chemical structures using the Markush DARC software," *Journal of Chemical Information and Computer Sciences*, vol. 37, no. 1, pp. 43–53, 1997.

[157] M. P. O'Hara and C. Pagis, "The PHARMSEARCH database," *Journal of Chemical Information and Computer Science*, vol. 31, no. 1, pp. 59–63, 1991.

[158] U. Schochgrubler, "(Sub)structure searches in databases containing generic chemical-structure representations," *Online Review*, vol. 14, no. 2, pp. 95–108, 1990.

[159] N. R. Schmuff, "A comparison of the MARPAT and Markush DARC software," *Journal of Chemical Information and Computer Sciences*, vol. 31, no. 1, pp. 53–59, 1991.

[160] R. N. Wilke, "Searching for simple generic structures," *Journal of Chemical Information and Computer Sciences*, vol. 31, no. 1, pp. 36–40, 1991.

[161] J. D. Holliday and P. Willett, "The influence of the DARC project on chemoinformatics research at the university of sheffield," *L'Actualite chimique*, vol. 320-321, no. 320-21, pp. 45–50, 2008.

[162] D. Domine and M. Cedric, "Method for fast substructure searching in non-enumerated chemical libraries," Nov 2007. US Patent Application US 2007/0260583 A1, [Online], Available: http://www.freepatentsonline.com/y2007/0260583.html.

[163] P. Foggia, C. Sansone, and M. Vento, "A performance comparison of five algorithms for graph isomorphism," in *In Proceedings of the 3rd IAPR TC-15 Workshop on Graph-based Representations in Pattern Recognition*, pp. 188–199, 2001.

[164] P. Foggia, C. Sansone, and M. Ventopatent, "A database of graphs for isomorphism and sub-graph isomorphism benchmarking," in *CoRR*, pp. 176–187, 2001.

[165] Daylight SMARTS examples; Daylight Chemical Information Systems, Inc.: Laguna Niguel, CA; http://www.daylight.com/dayhtml_tutorials/languages/smarts/smarts_examples.html, Accessed May 25, 2010.

[166] S. F. B. Abolmaali, J. K. Wegner, and A. Zell, "The compressed feature matrix - a fast method for feature based substructure search," *Journal of Molecular Modeling*, vol. 9, pp. 235–241, 2003. 10.1007/s00894-003-0126-0.

[167] M. Olah, C. Bologa, and T. I. Oprea, "An automated PLS search for biologically relevant QSAR descriptors," *Journal of Computer-Aided Molecular Design*, vol. 18, pp. 437–449, 2004. 10.1007/s10822-004-4060-8.

[168] P. Maass, T. Schulz-Gasch, M. Stahl, and M. Rarey, "Recore: A fast and versatile method for scaffold hopping based on small molecule crystal structure conformations," *Journal of Chemical Information and Modeling*, vol. 47, no. 2, pp. 390–399, 2007. PMID: 17305328.

[169] D. K. Agrafiotis, A. C. Gibbs, F. Zhu, S. Izrailev, and E. Martin, "Conformational sampling of bioactive molecules: A comparative study," *Journal of Chemical Information and Modeling*, vol. 47, no. 3, pp. 1067–1086, 2007. PMID: 17411028.

[170] J. Gasteiger and T. Engel, eds., *Chemoinformatics: A Textbook*. Wiley-VCH, 1 ed., Dec. 2003.

[171] L. C. Blum and J.-L. Reymond, "970 million druglike small molecules for virtual screening in the chemical universe database GDB-13," *Journal of the American Chemical Society*, vol. 131, no. 25, pp. 8732–8733, 2009.

[172] G. Schneider, W. Neidhart, T. Giller, and G. Schmid, "'Scaffold-hopping' by topological pharmacophore search: A contribution to virtual screening," *Angewandte Chemie (International ed. in English)*, vol. 38, no. 19, pp. 2894–2896, 1999.

[173] G. Schneider, O. Clment-Chomienne, L. Hilfiger, P. Schneider, S. Kirsch, H.-J. Bhm, and W. Neidhart, "Virtual screening for bioactive molecules by evolutionary de novo design special thanks to neil r. taylor for his help in preparation of the manuscript," *Angewandte Chemie (International ed. in English)*, vol. 39, no. 22, pp. 4130–4133, 2000.

[174] M. Hartenfeller, E. Proschak, A. Schller, and G. Schneider, "Concept of combinatorial de novo design of drug-like molecules by particle swarm optimization," *Chemical Biology and Drug Design*, vol. 72, no. 1, pp. 16–26, 2008.

[175] M. Hartenfeller, H. Zettl, M. Walter, M. Rupp, F. Reisen, E. Proschak, S. Weggen, H. Stark, and G. Schneider, "DOGS: Reaction-driven de novo design of bioactive compounds," *PLoS Computational Biology*, vol. 8, no. 2, p. e1002380, 2012.

[176] A. C. Good and R. A. Lewis, "New methodology for profiling combinatorial libraries and screening sets: cleaning up the design process with HARPick," *Journal of Medicinal Chemistry*, vol. 40, no. 24, pp. 3926–3936, 1997.

[177] V. J. Gillet, P. Willett, P. J. Fleming, and D. V. S. Green, "Designing focused libraries using MoSELECT," *Journal of Molecular Graphics and Modelling*, vol. 20, no. 6, pp. 491–498, 2002.

[178] C. Detering, H. Claussen, M. Gastreich, and C. Lemmen, "Knowledgespace - a publicly available virtual chemistry space," *Journal of Cheminformatics*, vol. 2, no. Suppl 1, pp. 1–9, 2010.

[179] N. Chomsky, "Systems of syntactic analysis," *Journal of Symbolic Logic*, vol. 18, no. 3, pp. 242–256, 1953.

[180] H. M. Berman, T. Battistuz, T. N. Bhat, W. F. Bluhm, P. E. Bourne, K. Burkhardt, Z. Feng, G. L. Gilliland, L. Iype, S. Jain, P. Fagan, J. Marvin, D. Padilla, V. Ravichandran, B. Schneider, N. Thanki, H. Weissig, J. D. Westbrook, and C. Zardecki, "The protein data bank," *Acta Crystallographica, Section D: Biological Crystallography*, vol. 58, no. Pt 6 No 1, pp. 899–907, 2002.

[181] P. J. Chase, "Algorithm 382: combinations of M out of N objects [G6]," *Communications of the ACM*, vol. 13, no. 6, pp. 368–, 1970.

[182] P. Maass, *Neue rechnergestützte Methoden zum Scaffold-Hopping unter Verwendung von Kristallstrukturen kleiner Molekule*. PhD thesis, University of Hamburg, Center for Bioinformatic, Computational Molecular Design, 2009.

[183] D. Weininger, "SMILES, a chemical language and information system. 1. introduction to methodology and encoding rules," *Journal of Chemical Information and Computer Science*, vol. 28, no. 1, pp. 31–36, 1988.

[184] S. Urbaczek, A. Kolodzik, J. R. Fischer, T. Lippert, S. Heuser, I. Groth, T. Schulz-Gasch, and M. Rarey, "NAOMI: On the almost trivial task of reading molecules from different file formats," *Journal of Chemical Information and Modeling*, vol. 51, no. 12, pp. 3199–3207, 2011.

A Publications

[A1] Hans-Christian Ehrlich and Matthias Rarey. Systematic benchmark of substructure search in molecular graphs - from Ullmann to VF2. *Journal of Cheminformatics*, 4(13), 2012.

The author of this thesis, H.-C. Ehrlich, implemented the presented software components, collected the data and performed the comparison studies. K. Schomburg help collecting the data. A. M. Henzler revised the manuscript. M. Rarey supervised the project.

[A2] Hans-Christian Ehrlich, Andrea Volkamer, and Matthias Rarey. Searching for substructures in fragment spaces. *Journal of Chemical Information and Modeling*, 52(12):3181–3189, 2012.

The author of this thesis, H.-C. Ehrlich, developed, implemented and tested the algorithm to search for chemical patterns in fragment spaces. In collaboration, A. Volkamer and H.-C.Ehrlich designed the use-cases and wrote the manuscript. M. Rarey supervised the work.

[A3] Hans-Christian Ehrlich, Angela M. Henzler, and Matthias Rarey. Searching for recursively defined generic chemical patterns in nonenumerated fragment spaces. *Journal of Chemical Information and Modeling*, 53(7):1676–1688, 2013.

The method to search for substructures in fragment spaces was extended by the author of this thesis, H.-C. Ehrlich, to allow a search for generic chemical pattern search using the SMARTS language. In collaboration, H.C. Ehrlich and A. M. Henzeler wrote the manuscript. M. Rarey revised supervised this work.

[A4] Karen Schomburg, Hans-Christian Ehrlich, Katrin Stierand, and Matthias Rarey. From structure diagrams to visual chemical patterns. *Journal of Chemical Information and Modeling*, 50(9):1529–1535, 2010.

The concept to visualize SMARTS pattern was collaboratively developed by K. Schomburg, K. Stierand and the author of this thesis, H.-C. Ehrlich. K. Schomburg implemented and evaluated the method. K. Stierand provided the graphical layout framework. H.-C. Ehrlich implemented a context-free grammar to parse SMARTS expressions in a data structure from which the semantic information was obtained. M. Rarey supervised this work.

[A5] Christin Schrfer, Tanja Schulz-Gasch, Hans-Christian Ehrlich, Wolfgang Guba, Matthias Rarey, and Martin Stahl. Torsion angle preferences in druglike chemical space: A comprehensive guide. *Journal of Medicinal Chemistry*, 56(5):2016–2028, 2013.

C. Schärfer developed designed and implemented the concept to analyze the conformational space of small molecules using SMARTS patterns. Together with T. Schulz-Gasch, C. Schärfer designed the SMARTS pattern and evaluated the method. The author of this thesis, H.-C. Ehrlich, developed and implemented a context-free grammar to parse expressions in the SMARTS language and implemented a subgraph isomorphism algorithm to match patterns to molecules. W. Guba, M. Stahl and M. Rarey supervised the project and revised the manuscript.

[A6] Hans-Christian Ehrlich and Matthias Rarey. Maximum common subgraph isomorphism algorithms and their applications in molecular science: A review. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 1(1):68–79, 2011.

The current literature on the maximal common subgraph isomorphism algorithms and their application on molecular data was reviewed by the author of this thesis, H.-C. Ehrlich. A. M. Henzeler revised the manuscript. M. Rarey supervised the work.

B Conferences

[B1] Oral presentation at 9th International Conference on Chemical Structures 2011, Noordwijkerhout, The Netherlands: *On the exclusion of unwanted chemical patterns from large fragment spaces*

[B2] Poster presentation at 6th German Conference on Chemoinformatics 2010, Goslar, Germany: *Searching for substructures in fragment spaces*

[B3] Poster presentation at 5th Sheffield Conference on Chemoinformatics 2010, Sheffield, United Kingdom: *Searching substructures in fragment spaces*

[B4] Participation at 2nd Strasbourg Summer School on Chemoinformatics 2010, Strasbourg, France

[B5] Participation at 1st Many-Core and Reconfigurable Supercomputing Conference 2009, Berlin, Germany

[B6] Representative for the University of Hamburg at Biotechnica 2009, Hannover, Germany

C Supplementary information

C.1 Chemical pattern search in fragment spaces

This section supplies additional information on algorithms used in *SubSubSearch* that exceeded the scope of publications [A1], [A2] and [A3].

C.1.1 SMARTS grammar

The SMARTS language is used to describe chemical patterns throughout this thesis and in all publications. In order to translate a SMARTS string expression into a computer-processable structure a context-free grammar was developed that defines the syntax and detects the semantics of SMARTS expressions. Table C.1, C.2 and C.3 describe the grammar. In addition to the language specifications defined by Daylight [37], this grammar supports SYBYL types, e.g., '{C.3}', amide bonds '-^', hybridization states '*<number>', link atoms '[?<name>?]' or '[*<number>]', nitrogens with an attached lone pair 'n_lp', and a smaller relation '<' for the charge. Note, that the usage of Sybyl types is not supported by the matching routine for SMARTS and will cause an error.

Even though the SMARTS language is relatively well defined to be modeled as context-free grammar, some problems are known. The language allows the use of chemical element abbreviation, e.g., 'C' for a carbon. The abbreviation can lead to an ambiguous grammar, e.g, the abbreviation for strontium is 'Sr'. A SMARTS expression describing a strontium atom is '[Sr]'. That expression can also be interpreted as an aliphatic sulfur 'S' that is present in at least one ring 'r'. The element abbreviation for strontium 'Sr', chromium 'Cr', rhodium 'Rh', radon 'Rn', radium 'Ra', and praseodymium 'Pr' were therefore excluded. Fortunately, these elements rarely occur in organic chemistry and are therefore of minor relevance in the field of drug discovery.

Rule	**Production**
input:	smarts *or* reaction
reaction:	smarts reactionbond smarts
smarts:	grouping *or* branch_open disconnected branch_close *or* branch_open atomconnection branch_close *or* atomconnection
grouping:	grouping disconnectedbond atomconnection *or* grouping disconnectedbond branch_open atomconnection branch_close *or* atomconnection disconnectedbond atomconnection *or* branch_open atomconnection branch_close disconnectedbond branch_open atomconnection branch_close
disconnected:	disconnected disconnectedbond atomconnection *or* atomconnection disconnectedbond atomconnection
recursion:	recursion_start branch_open smarts branch_close
atomconnection:	atomconnection bondchain atom *or* atomconnection bondchain atom_ring *or* atomconnection bondchain integer *or* atomconnection atom *or* atomconnection branch *or* atomconnection atom_ring *or* atomconnection integer *or* atom
atom:	bracket_open atompropchain bracket_close *or* symbol
atompropchain:	atomprop_lowand
atomprop_lowand:	atomprop_lowand low_and atomprop_or *or* atomprop_lowand labelexpr *or* atomprop_or
atomprop_or:	atomprop_or or atomprop_highand *or* atomprop_or or massexpr atomprop_highand *or* atomprop_highand *or* massexpr atomprop_highand
atomprop_highand:	atomprop_highand high_and atompropexpr *or* atomprop_highand atompropexpr *or* atompropexpr
atompropexpr:	not atomprop *or* atomprop
labelexpr:	label integer
atom_ring:	ring_identifier integer
atomprop:	degree *or* degree integer *or* implicit_hydrogen *or* implicit_hydrogen integer *or* hydrogen *or* hydrogen integer *or* ring_member *or* ring_member integer *or* ring_size *or* ring_size integer *or* valence *or* valence integer *or* connectivity *or* connectivity integer *or* ring_connectivity *or* ring_connectivity integer *or* charge *or* charge digit *or* smaller charge *or* smaller charge digit *or* chiral *or* chiral integer *or* chiral maybe *or* chiral chiralclass *or* chiral chiralclass maybe *or* chiral chiralclass integer *or* chiral chiralclass integer maybe *or* atomnumber integer *or* hybridisation integer *or* isolink integer *or* link *or* nlp *or* symbol *or* recursion

Table C.1: Part 1 of SMARTS context-free grammar.

symbol:	aliphatic *or* aromatic *or* sybyl *or* atomwildcard
massexpr:	not mass *or* mass
mass:	integer
charge:	negcharge *or* poscharge
branch:	branch_open bondchain atomconnection branch_close *or* branch_open atomconnection branch_close
bondchain:	bond_lowand
bond_lowand:	bond_lowand low_and bond_or *or* bond_or
bond_or:	bond_or or bond_highand *or* bond_highand
bond_highand:	bond_highand high_and bondexpr *or* bond_highand bondexpr *or* bondexpr
bondexpr:	not bond *or* bond
bond:	singlebond *or* doublebond *or* triplebond *or* aromaticbond *or* amidbond *or*
	upbond *or* upbond maybe *or* downbond *or* downbond maybe *or* anyringbond *or* anybond
aliphatic:	ALIPHATIC_WILDCARD='A' *or* ALIPHATIC=('B' *or* 'C' *or* 'N' *or* 'O' *or* 'P' *or* 'S' *or* 'F' *or* 'Cl' *or* 'Br' *or* 'I' *or* 'Si' *or* 'As' *or* 'Se' *or* 'Te' *or* 'Li' *or* 'Na' *or* 'K' *or* 'Rb' *or* 'Cs' *or* 'Be' *or* 'Mg' *or* 'Ca' *or* 'Ba' *or* 'Sc' *or* 'U' *or* 'La' *or* 'Ti' *or* 'Zr' *or* 'Hf' *or* 'V' *or* 'Nb' *or* 'Ta' *or* 'Mo' *or* 'W' *or* 'Mn' *or* 'Tc' *or* 'Re' *or* 'Fe' *or* 'Ru' *or* 'Os' *or* 'Co' *or* 'Ir' *or* 'Ni' *or* 'Pd' *or* 'Pt' *or* 'Cu' *or* 'Ag' *or* 'Au' *or* 'Zn' *or* 'Cd' *or* 'Hg' *or* 'Al' *or* 'Ga' *or* 'In' *or* 'Tl' *or* 'Ge' *or* 'Sn' *or* 'Pb' *or* 'Sb' *or* 'Bi' *or* 'Po' *or* 'At' *or* 'Fr' *or* 'Ac' *or* 'Ce' *or* 'Nd' *or* 'Pm' *or* 'Sm' *or* 'Eu' *or* 'Gd' *or* 'Tb' *or* 'Dy' *or* 'Ho' *or* 'Er' *or* 'Tm' *or* 'Yb' *or* 'Lu' *or* 'Th' *or* 'Pa' *or* 'U' *or* 'Np' *or* 'Pu' *or* 'Am' *or* 'Cm' *or* 'Bk' *or* 'Cf' *or* 'Es' *or* 'Fm' *or* 'Md' *or* 'No' *or* 'Lr')
aromatic:	AROMATIC_WILDCARD='a' *or* AROMATIC=('b' *or* 'c' *or* 'n' *or* 'o' *or* 'p' *or* 's' *or* 'si' *or* 'as' *or* 'se' *or* 'te')
atomnumber:	POUND='#'
atomwildcard:	STAR='*'
chiral:	ATSIGN='@' *or* ATATSIGN='@@'
sybyl:	SYBYL=('C.3 ' *or* 'C.2 ' *or* 'C.1 ' *or* 'C.ar ' *or* 'C.cat ' *or* 'N.3 ' *or* 'N.2 ' *or* 'N.1 ' *or* 'N.ar ' *or* 'N.am ' *or* 'N.pl3 ' *or* 'N.4 ' *or* 'O.3 ' *or* 'O.2 ' *or* 'O.co2 ' *or* 'O.spc ' *or* 'O.t3p ' *or* 'S.3 ' *or* 'S.2 ' *or* 'S.O ' *or* 'S.O2 ' *or* 'P.3 ' *or* 'F ' *or* 'Cl ' *or* 'Br ' *or* 'I ' *or* 'H ' *or* 'H.spc ' *or* 'H.t3p ' *or* 'LP ' *or* 'Du ' *or* 'Du.C ' *or* 'Any ' *or* 'Hal ' *or* 'Het ' *or* 'Li ' *or* 'Na ' *or* 'Mg ' *or* 'Al ' *or* 'Si ' *or* 'K ' *or* 'Ca ' *or* 'Cr.th ' *or* 'Cr.oh ' *or* 'Mn ' *or* 'Fe ' *or* 'Co.oh ' *or* 'Cu ' *or* 'Zn ' *or* 'Se ' *or* 'Mo ' *or* 'Sn')
integer:	INTEGER='[0-9]+'
digit:	DIGIT='[0-9]+.[0-9]+' *or* INTEGER='[0-9]+'
recursion_start:	RECURSION_START='$'
branch_open:	BRANCH_OPEN='('
branch_close:	BRANCH_CLOSE=')'
singlebond:	MINUS='-'
doublebond:	DOUBLEBOND='='
triplebond:	POUND='#'
aromaticbond:	AMIDBOND='-^'
anyringbond:	ATSIGN='@'
anybond:	ANYBOND='~'

Table C.2: Part 2 of SMARTS context-free grammar.

upbond:	UPBOND='/'
downbond:	DOWNBOND='\'
chiralclass:	THETRAHEDRAL=TH" *or* ALLENELIKE='AL' *or* SQUAREPLANAR='SP' *or* TRIGONALBOPYRAMIDAL='TB' *or* OCTAHEDRAL='OH'
degree:	DEGREE='D'
implicit_hydrogen:	IMPLICIT_HYDROGEN='h'
hydrogen:	HYDRO='H'
ring_member:	RING_MEMBER='R'
ring_size:	RING_SIZE='r'
ring_identifier:	RING_IDENTIFIER='%'
valence:	VALENCE='v'
connectivity:	CONNECTIVITY='X'
ring_connectivity:	RING_CONNECTIVITY='x'
negcharge:	MINUS negcharge *or* MINUS='-'
poscharge:	POS_CHARGE poscharge *or* POS_CHARGE='+'
label:	COLON=':'
high_and:	HIGH_AND='&'
low_and:	LOW_AND=';'
or:	OR=','
not:	NOT='!'
maybe:	MAYBE='?'
bracket_open:	BRACKET_OPEN='['
bracket_close:	BRACKET_CLOSE=']'
disconnectedbond:	DISCONNECTEDBOND='.'
reactionbond:	REACTION='>>'
hybridisation:	HYBRIDISATION='^'
link:	LINK='?'
nlp:	NLP='n_lp'
isolink:	STAR='*'
smaller:	SMALLER='<'

Table C.3: Part 3 of SMARTS context-free grammar.

C.1.2 Bi-connected component algorithm

In publication [A2] an algorithm is used to calculate the *bi-connected components* (BCCs) of a graph. The BCCs are used to separate a graph representing a chemical pattern into subgraphs or sub-patterns in which cyclic parts are conserved. In that tree, nodes represent the components and an edge connects two components if and only if the components share an *articulation point* that is a common node. The BCC of a graph $G = (V, E)$ is a subgraph $G' = (V', E')$ such that $V' \subset V$ and $E' \subset E$ that is connected even if one edge $e \in E'$ is removed. The algorithm shown in Algorithm 1 and 2 follows the idea of Tarjan to calculate BCCs [73]. Tarjan's algorithm employs a depth-first search strategy that records the depth d of each node and the *low point* low that is the lowest recorded depth of all descendants. The key to compute BCCs from assigned depth and low points is that any non-root node v for which a child node u exists such that $low[u] \geq d[v]$ separates two BCCs. This condition to identify articulation points is checked after the depth and low points for each descendant are calculated. Therefore, the algorithm assigns edges to different BCCs.

Algorithm 1 Calculate bi-connected components of a graph.

```
1:  input graph G = (V, E)
2:  output array bcc assigning each edge to a BCC (each index represents one edge, array entry
    is the edge's BCC)
3:  procedure CalcBCCs(G = (V, E))
4:      time ← 0
5:      bccNum ← 0
6:      stack ← ∅
7:      for all v ∈ V do
8:          d[v] ← 0                                  ▷ discovery time
9:          low[v] ← 0                                ▷ low point
10:     for all e ∈ E do
11:         bcc[e] ← 0                                ▷ bi-connect component number
12:     for all v ∈ V do
13:         if d[v] = 0 then
14:             CalcBCCsVisit( &v, &bccNum, &stack )
15:     return G
```

Algorithm 2 Recursive sub-routine of CalcBCCs.

```
1: input node u, BCC number bccNum, stack S
2: output array bcc assigning each edge to a BCC (each index represents one edge, array entry
   is the edge's BCC)
3: procedure CalcBCCsVisit(v, *bccNum, *S)
4:     time ← time + 1
5:     d[v] ← time
6:     low[v] ← d[v]
7:     for all uv ∈ Adj[v] do
8:         if d[u] <d[v] then
9:             push(S, e = (v, u))                    ▷ discovered back-edge
10:        if d[u] = 0 then                           ▷ tree edge
11:            CalcBCCsVisit( v, bccNum, S )
12:            low[v] ← min(low[v], low[u])           ▷ minimum value over all descendants
13:            if low[u] ≥ d[v] then                  ▷ new BCC discovered
14:                repeat                             ▷ assign edges to bi-connected component
15:                    e ← pop(S)
16:                    bcc[e] ← bccNum
17:                until e ≠ (v, u)
18:                bccNum gets bccNum + 1
19:        else                                       ▷ undiscovered back-edge
20:            low[v] gets min(low[v],low[u])         ▷ minimum value over all back-edges
```

C.1.3 Subgraph or sub-pattern enumeration algorithm

In order to enumerate subgraphs, the BCC algorithm is used to generate a BCC tree. As input for the algorithm an ordered BCC tree is represented as array in which the array index represents the BCC tree nodes and an entry the corresponding parent node. In order to enumerate all connected subgraphs, the BCC tree must be depth-first search (DFS) ordered starting at an arbitrary node. A 'NULL' entry in the input array marks the root. Algorithm 3 and 4 show the enumeration algorithm. The algorithm follows a strategy to enumerate all subsets based on a lexicographical order [181, 182]. The DFS order of components is essential such that no subgraphs are missed and all enumerated subgraphs are connected. The condition to ensure the complete enumeration of all subgraphs is that no component of higher order connects two components of lower order. In addition, the algorithm avoids the enumeration of disconnected subgraphs by only removing components such that a component of higher order is not removed when the removal separates two components of lower order. In each iteration, the algorithm choses the highest order component that is part of the current subgraph as root. New components i are chosen top-down the order until a component is found that is smaller than the root or that is connected to the current subgraph. Components of higher order than i are removed. Figure C.1 shows an enumeration example. The strategy was adapted from Mass [182].

Algorithm 3 Enumerate all connected subgraphs of a graph.

```
1: input array describing the BCC tree of a graph
2: output arrays indicating which BCC nodes belong to a subgraph
3: procedure EnumerateSubgraphs(A)
4:     allSubgraph ← ∅
5:     nextSubgraph gets Array[length[A]]          ▷ create the first subgraph array
6:     for all a ∈ nextSubgraph do
7:         a ← 0
8:     finalSubgraph gets Array[length[A]]         ▷ create the final subgraph array
9:     for all b ∈ finalSubgraph do
10:        b ← 1
11:    while nextSubgraph ≠ finalSubgraph do
12:        nextSubgraph = calcNextSubgraph(nextSubgraph)
13:        push(allSubgraph, nextSubgraph)
14:    return allSubgraphs
```

Algorithm 4 Subroutine of EnumerateSubgraphs to calculate the next subgraph.

```
1: input array of the previous subgraph (chosen components have a non-zero entry)
2: output arrays of the next subgraph
3: procedure calcNextSubgraph(lastSubgraph)
4:     nextSubgraph gets lastSubgraph
5:     nextSgLength = length[nextSubgraph] - 1
6:     rootIdx = nextSgLength
7:     for i ← nextSgLength → 0 do                 ▷ determine index of root node
8:         if nextSubgraph[i] = 1 then
9:             rootIdx = i
10:    for i ← nextSgLength → 0 do
11:        if nextSubgraph[i] = 0 then             ▷ component i is not chosen
12:            if root < i and nextSubgraph[A[i]] = 0 then   ▷ component i can not be a root of a new subgraph and the parent of i is not part of the current subgraph
13:                continue                        ▷ component i can not be part of the subgraph
14:            nextSubgraph[i] ← 1                 ▷ add component i to subgraph
15:            for j ← i + 1 → length[nextSubgraph] do
16:                nextSubgraph[j] ← 0             ▷ remove all components > i
17:            return nextSubgraph
```

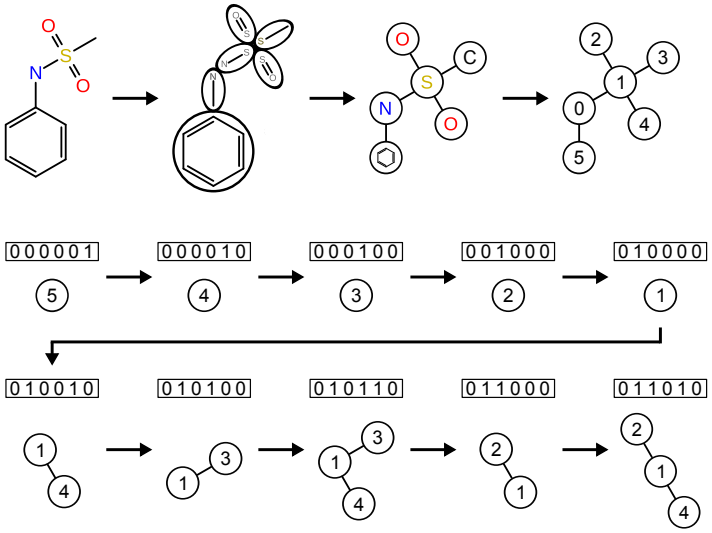


Figure C.1: Enumeration of a subgraph. The algorithm converts the graph into a depth-first search ordered bi-connected component (BCC) tree (top). Below the conversion of the first ten enumeration step are shown. A bit array indicates with a non-zero entry which BCCs are chosen. From the previous subgraph the next subgraph is calculated as described by Algorithm 3 and Algorithm 4.

C.2 Chemical pattern exclusion from fragment spaces

The Figures C.2, C.3 and C.4 depict the PAINS patterns used in the exclusion experiments presented in Section 4.3.

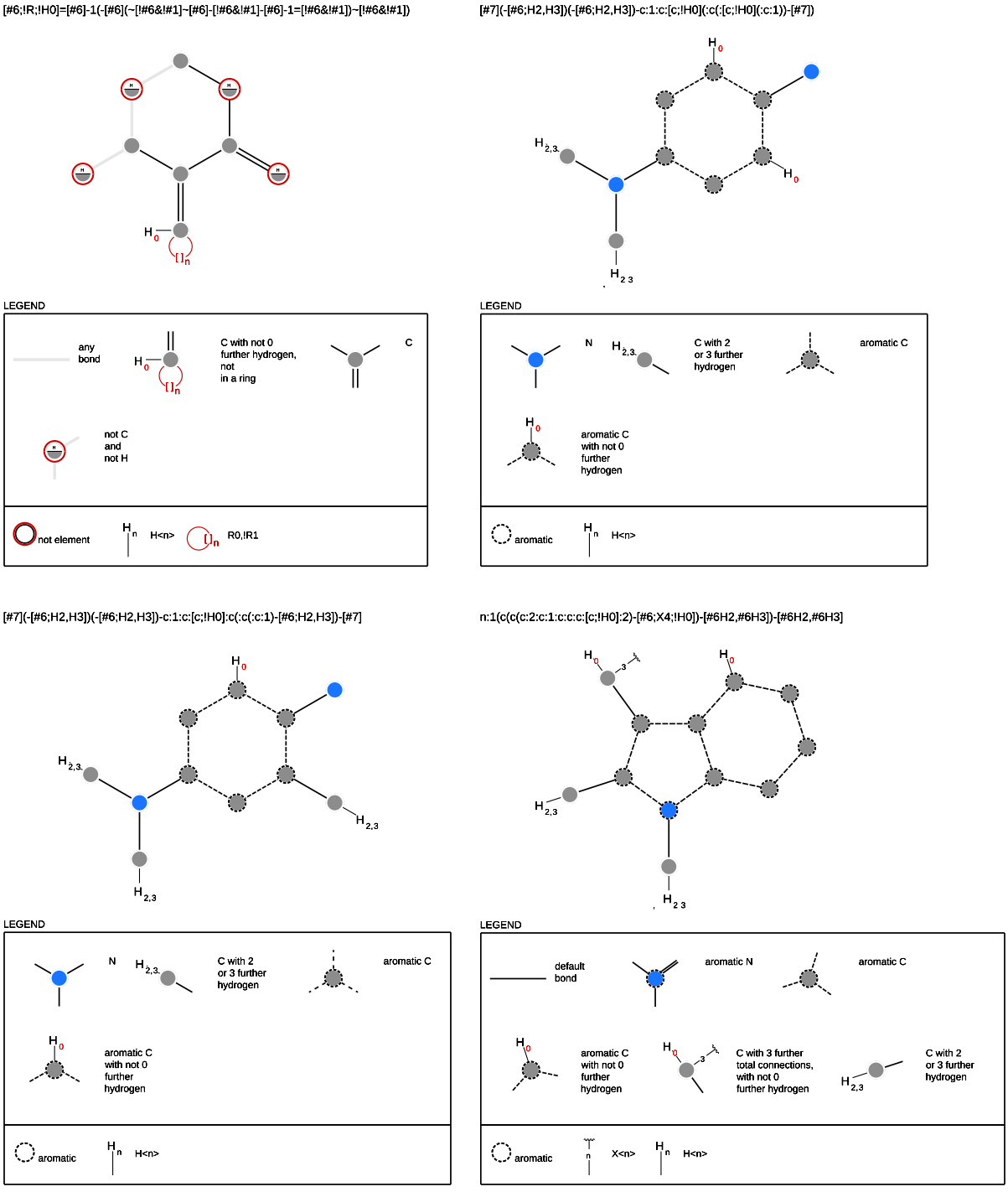
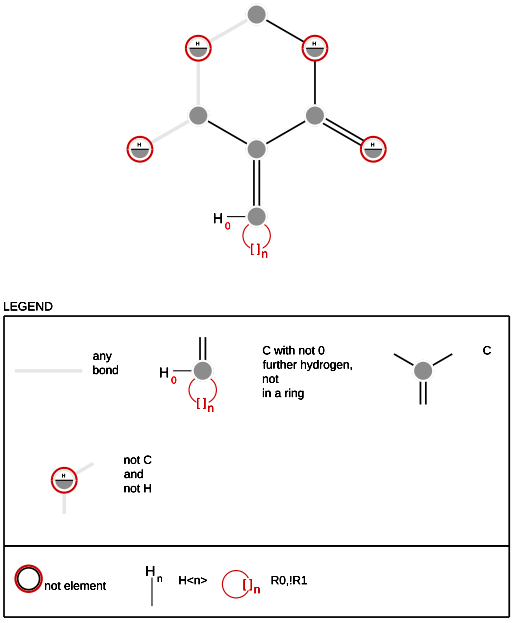
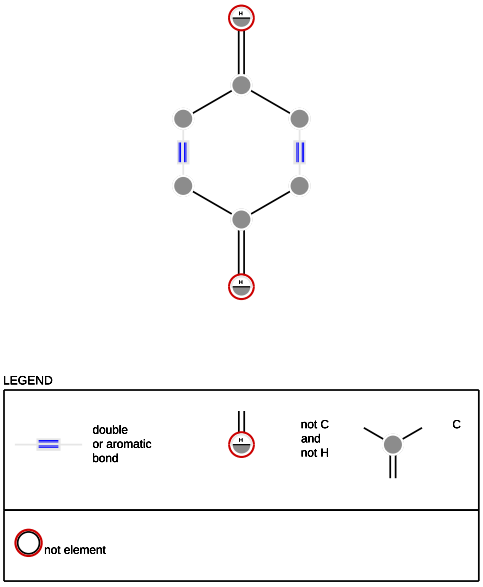


Figure C.2: Visualization of PAINS 1-4 SMARTS pattern (top-left to bottom-right).

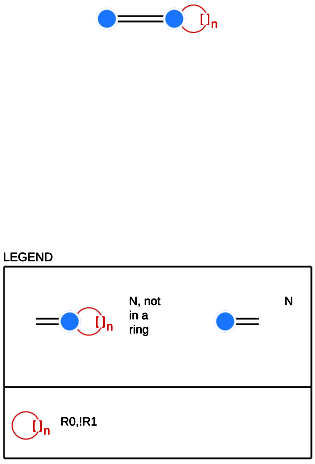
[#6;!R;!H0]=[#6]-1(-[#6](~[!#6&!#1]~[#6]-[!#6&!#1]-[#6]-1=[!#6&!#1])~[!#6&!#1])



[!#6&!#1]=[#6]-1-[#6]=,:[#6]-[#6](=[!#6&!#1])-[#6]=,:[#6]-1



[#7;!R]=[#7]



[#8;!H0]-c:1:c:c:c:c:c:1-[#6;X4]-[#7]

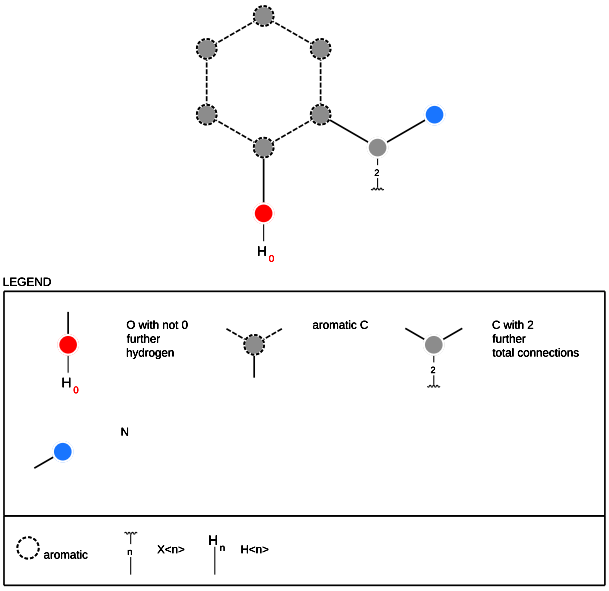


Figure C.3: Visualization of PAINS 5-8 SMARTS pattern (top-left to bottom-right).

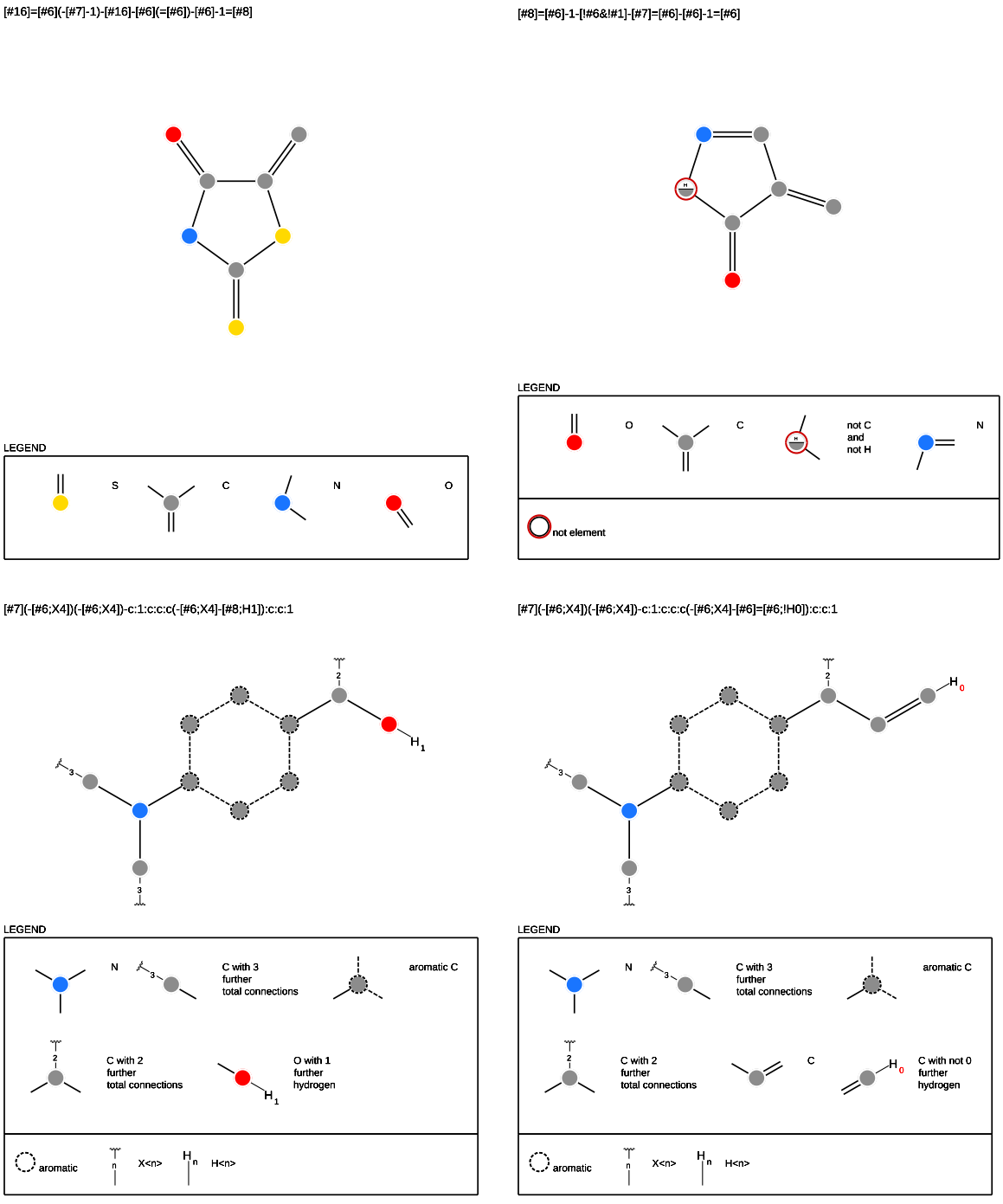


Figure C.4: Visualization of PAINS 9-12 SMARTS pattern (top-left to bottom-right).

D Developer & User information

This section provides information regarding the implementation and user interface of the SmartsFs software library which is part of the Flex software suit. The library functionalities are also usable through a command-line tool.

D.1 About SmartsFs library

The SmartsFs library is written in C/C++ and provides the algorithms and data structures to search for SMARTS patterns in fragment spaces. The algorithm avoids a costly enumeration of molecules during the search and is designed to minimize the number of explored link connections. In general, the algorithm follows a four step procedure: The query SMARTS pattern is separated into sub-patterns (SPs) in which cyclic parts are conserved and missing pattern parts are indicated with linker nodes. These SPs are searched inside fragments assuring that linker nodes are only assigned to link atoms. For each SP a list of matching fragments is recorded. These lists are combined to form a set of recombination trees that describe how SPs are connected to form the original pattern and how fragments need to be assembled to obtain products that contain the query pattern. From the recombination trees, products or molecule are enumerated by connecting the fragments according to a tree's topology. The algorithm supplies products or molecules that represent a minimal connection of fragments such that they include the query pattern. Products may contain open linkers that allow to attach further fragments. Molecules are products in which all open linkers are saturated with terminal groups.

The search times range from seconds to minutes for small and complex fragment spaces. In rare cases, a runtime of several days are possible. The enumeration of products or molecules needs a few milliseconds per product/molecule.

D.2 SmartsFs library organization

The following section describes the dependencies of the SmartsFs library on other libraries of the Flex software suit. The libraries often support a large spectrum of functionality, here, only the functionality used by SmartsFs is mentioned.

SmartsFs depends on the fragment space library, the molecule database library and the SMARTS library. The dependencies are shown in Figure D.1. The fragment space (FragSpace) library provides the functionality to read fragment spaces from file and to handle fragments and connection rules as needed by SmartsFs. The molecule database (MolDb) library provides the functionality to store molecules and fragments out-of-memory and to compare fragments by their unique SMILES identifier. These features are essential during the enumeration of products from recombination trees. Often the number of products is too large to be kept in main memory for a unique SMILES comparison. The SMARTS library handles the conversion

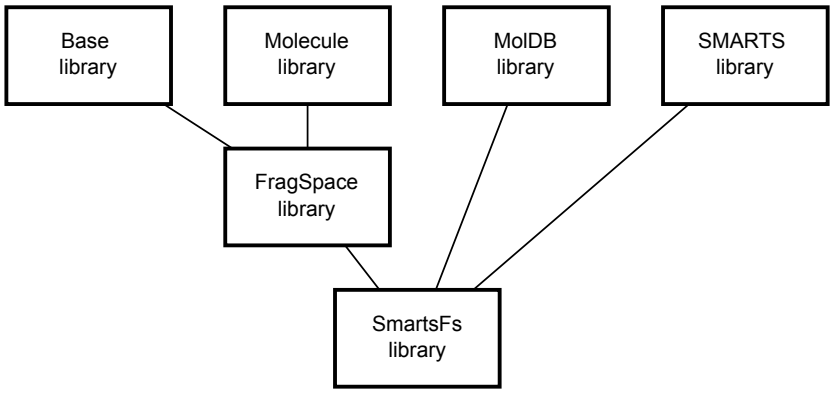


Figure D.1: Internal SmartsFs library dependencies.

of SMARTS strings into advanced data structures and provides algorithms to search for patterns in molecules. This search is used by SmartsFs during the search for SPs in fragments. The Base library is used in the fragment space library and also in SmartsFs (connection not shown). In general, it supplies functions and macros to handle the amount of information displayed during the software runs.

D.3 Dependencies on external libraries and programs

SubSubSearch uses functions and data structures from the external Boost and Qt libraries. The SMARTS grammar depends on the bison/flex software for context-free grammars. The libraries as well as the bison/flex software must be available when compiling the SmartsFs library.

D.4 Important limitations

At the current stage, the search algorithm has some limitations regarding terminal groups, hydrogen substitution patterns, stereoisomeric structures and products constructed by a cyclic connection of fragments. The search algorithm separates the query pattern such that cyclic parts are conserved which allows a tree-like connection of fragments. Therefore, macro-cyclic structures that require a connection of fragments into cycles are not found. Terminal groups are used to saturate open valences of products converting them into complete molecules. These terminal groups are not considered during the search. When terminal groups are a hydrogen atom, the algorithm may mismatch nodes that describe atoms with a specific hydrogen substitution pattern. For example, an atom of a fragment that has two hydrogens and one link atom attached, can have three hydrogens after termination. The change in hydrogen substitution pattern is not addressed during the search. Another problem regarding the attachment at open valences arises when stereoisomeric structures are of interest. The assignment of stereo centers in products is impossible, since an attachment of further fragments might result in a symmetric product which causes the stereo center to vanish. Stereo centers are

therefore not addressed during the search and must be checked after the construction of complete molecules in a post-processing step. Additional strategies to circumvent these problems are described in Section 4.2.

D.5 Search and enumeration interfaces

The interface to the search algorithm in SmartsFs is a single function of the form `SubSubMatcher( sg, fspace, maxNofFrags, premapping, uinfo, explicitH )`. The parameter `sg` is a subgraph structure obtained from parsing a SMARTS pattern using the SMARTS library. Parameter `fspace` is a fragment space that is obtained from a fragment space fsf-file using the FragSpace library. The maximal number of fragments a product is composed of is controlled by the parameter `maxNofFrags`. The user info supplied by `uinfo` allows to control the amount of information output during a run of the `SubSubMatcher` function. The matcher differentiates between an implicit hydrogen matching `explicitH=false` and an explicit hydrogen matching `explicitH=true`. When hydrogens are implicitly matched, SMARTS language symbols for generic atoms such as '*' for any atom or 'A' for any aliphatic atom are only assigned to heavy atoms. In contrast, these generic symbols are also assigned to hydrogen atoms when an explicit matching of hydrogens is activated. When a SMARTS pattern includes explicitly defined hydrogen atoms, e.g, 'C=C([H])[H]', the parameter `explicitH` must be set `true`, otherwise no matches are found. The function returns a solution in the form of recombination trees describing products that include the query pattern. The set of fragments described by each recombination tree is minimal in the sense that a combination of these fragments result in a product that include the query pattern and often have open valences where further fragments can be attached. The attachment of further fragments obviously results in a different product that also includes the pattern. The interface for the enumeration of products from a recombination tree is `smfsSubSubEnumAsMolDb(sol, fspace, dbname, uinfo, maxNofProducts, minNofFrags, maxNofFrags)`. Parameter `sol` is a solution obtained from calling `SubSubMatcher`. The fragment space in which the pattern was searched is given by `fspace`. During the enumeration, the algorithm removes duplicate products. For that purpose, a molecule database of the MolDb library is utilized and temporarily stored at `dbname`. The information level is defined by `uinfo`. The parameters `maxNofProducts`, `minNofFrags` and `maxNofFrags` define the maximal number of enumerated products and the minimal and maximal number of fragments in a product. The default for these parameters is that all products described by the solution are enumerated with no limitation to their size.

D.6 Search tool

The command-line tool `SmartsFsMatcher` allows the search for patterns in fragment spaces. A command-line call has the form `SmartsFsMatcher <smarts_string/file><fsf-file><outfile><num_products>`. The command line

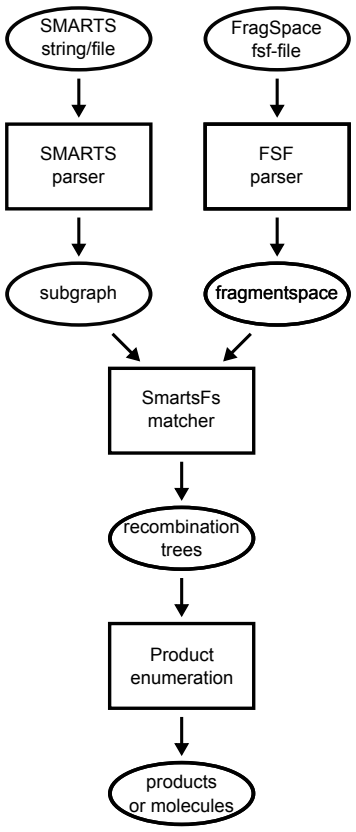


Figure D.2: Data flow in SmartsFs command-line tool.

parameters are `smarts_string` that is a SMARTS expression as string, `smarts_file` a file that holds multiple SMARTS string, `fsf-file` a fragment space file in fsf-format, `outfile` the prefix for the file to which the products and molecules are written, and `num_products` the number of products at which the enumeration is stopped. If `num_products` is supplied, the first `num_products` are enumerated. If `num_products` is not supplied, the tool enumerates all products. Figure D.2 shows the data flow.

D.7 File formats

The following section specifies the input and output formats for SMARTS strings and files, fragment spaces and the file output.

D.7.1 SMARTS input files

SMARTS strings must follow the SMARTS specification provided by Daylight [37]. The specification was extended to support SYBYL types, amid bonds, hybridisation states, link atoms, nitrogens with a lone pair, and smaller relations for the charge. These extensions are described in the SMARTS grammar given in Section C.1.1. A SMARTS file holds line-separated SMARTS strings followed by an space-separated identifier. The following shows an example.

```
#SMARTS   id
C-!@C     bajorath.smarts_0
C=@A[I]   bajorath.smarts_2
C-@C      bajorath.smarts_3
S-!@S     bajorath.smarts_4
a-[Cl]    bajorath.smarts_5
C=!@C     bajorath.smarts_6
S-@S      bajorath.smarts_7
```

D.7.2 Fragment space input files

Fragment spaces must follow the fsf-format. The following listing defines the format and shows part of the BRICS 4k fragment space [42] as an example. Supported molecule file types are MOL2, SDF and SMILES [183]. Details regarding the chemical model and the file types are described in [184].

```
# syntax description:
# @link_types <nof links>
#    <link name 1> <link name 2> ...
#    <link name k> <link name k+1> ... <link name <nof links>>
#
# @fragment_files <nof files>
#    < frag file 1>
#    < frag file 2>
#    :
#    < frag file <nof files>>
#
# @link_terminal_groups
#    <link name 1> <group> [<atom type> <bond type> <bond length> <torsion angle>]
#    <link name 2> <group> [<atom type> <bond type> <bond length> <torsion angle>]
#    :
#    <link name <nof links>> <group> [<atom type> <bond type> <bond length> <torsion angle>]
#
# @link_compatibility_matrix
#    <name of> <name of> [<aatom type> <aatom type>] <bond>  <bond>  [<torsion>]
#     link_1    link_2     for_link_1   for_link_2    type   length     angle
#    :

@link_types 16
  R1 R2 R3 R4 R5 R6 R7 R8 R9 R10 R11 R12 R13 R14 R15 R16

@fragment_files 1
  brics_2008_4k.mol2

@link_terminal_groups
```

```
  link   group   aatom   bond   blen   torsion
  R1     [CH3]   *       1      1.507  *
  R2     C(=O)C  *       1      1.337  180
  R3     [CH3]   *       1      1.429  *
  R4     [H]     *       1      1.090  *
  R5     [CH3]   *       1      1.398  *
  R6     [CH3]   *       1      1.507  *
  R7     [CH2]   *       2      1.316  180
  R8     [H]     *       1      1.090  *
  R9     [CH3]   *       1      1.465  *
  R10    [CH3]   *       1      1.465  *
  R11    [CH3]   *       1      1.815  *
  R12    [CH3]   *       1      1.816  *
  R13    [H]     *       1      1.080  *
  R14    [H]     *       1      1.080  *
  R15    [H]     *       1      1.080  *
  R16    [H]     *       1      1.080  *

@link_compatibility_matrix
  link1   link2   aatom1   aatom2   bond   blen   torsion
  R1      R2      *        *        1      1.355  180
  R1      R3      *        *        1      1.362  *
  R1      R10     *        *        1      1.355  180
  R2      R12     *        *        1      1.656  *
  R2      R14     *        *        1      1.398  *
  R2      R16     *        *        1      1.398  *
  R3      R4      *        *        1      1.452  *
  R3      R13     *        *        1      1.429  *
  R3      R14     *        *        1      1.362  *
  R3      R15     *        *        1      1.429  *
  R3      R16     *        *        1      1.362  *
# (following linkers not shown)
```

D.7.3 Output files

The prefix defined by `outfile` is used to generate two files: `outfile.products.smi` and `outfile.molecules.smi`. The file `outfile.products.smi` contains all enumerated products in SMILES format in the form <`SMILES string`><`identifier`> in each line. The file `outfile.molecules.smi` is in the same format and contains the corresponding molecules obtained by saturating the open valences at products with the corresponding terminal groups. Both files are in the same order and therefore products and corresponding molecules can be found in the same line number.

E Journal articles

A1 Systematic benchmark of substructure search in molecular graphs - From Ullmann to VF2

Authors:	Hans-Christian Ehrlich, Matthias Rarey
Type of publication:	Journal article
Reference:	Journal of Cheminformatics, 4(13), 2012, doi:10.1186/1758-2946-4-13
Status:	Published
Legal:	Open-access under Creative Commons Legal Code.

Journal of Cheminformatics

RESEARCH ARTICLE Open Access

Systematic benchmark of substructure search in molecular graphs - From Ullmann to VF2

Hans-Christian Ehrlich and Matthias Rarey*

Abstract

Background: Searching for substructures in molecules belongs to the most elementary tasks in cheminformatics and is nowadays part of virtually every cheminformatics software. The underlying algorithms, used over several decades, are designed for the application to general graphs. Applied on molecular graphs, little effort has been spend on characterizing their performance. Therefore, it is not clear how current substructure search algorithms behave on such special graphs. One of the main reasons why such an evaluation was not performed in the past was the absence of appropriate data sets.

Results: In this paper, we present a systematic evaluation of Ullmann's and the VF2 subgraph isomorphism algorithms on molecular data. The benchmark set consists of a collection of 1235 SMARTS substructure expressions and selected molecules from the ZINC database. The benchmark evaluates substructures search times for complete database scans as well as individual substructure-molecule pairs. In detail, we focus on the influence of substructure formulation and size, the impact of molecule size, and the ability of both algorithms to be used on multiple cores.

Conclusions: The results show a clear superiority of the VF2 algorithm in all test scenarios. In general, both algorithms solve most instances in less than one millisecond, which we consider to be acceptable. Still, in direct comparison, the VF2 is most often several folds faster than Ullmann's algorithm. Additionally, Ullmann's algorithm shows a surprising number of run time outliers.

Keywords: Substructure search, Subgraph isomorphism, Algorithm, Benchmark, SMARTS, Chemical pattern search

Background

Today's drug discovery faces a constantly growing number of commercially available or synthetically accessible compounds maintained in large databases [1,2]. In order to efficiently search such databases, computational search strategies comprising various search criteria have been developed over more than four decades [3-14]. Search criteria range from retrieving the one exact compound over selecting compounds via substructure features to the application of various similarity measures. In the following, we focus on methods that test compounds for the presence of certain functional groups or substructures.

Modeling molecular structures as labeled graphs has a long tradition and gives the basis for modern cheminformatics methods. A graph-based representation is chemically intuitive and forms a solid theoretical foundation for computer-aided processing. Furthermore, graphs allow the substructure search problem to be solved by graph isomorphism techniques, i.e., searching molecules for substructures is equivalent to testing two labeled graphs for subgraph isomorphism. The subgraph isomorphism problem is well studied [15-17] and one of the oldest and most applied algorithms [18-22] was introduced by Ullmann in 1976 [7]. Over the years that followed, only a few subgraph isomorphism methods were introduced [11,16,23], the most recent being the VF2 algorithm [12].

Until now, each comparison of (sub-)graph isomorphism algorithms [16,17] only employs synthetic graph data. The data is most often constructed to show the algorithms' behavior on medium to large graphs. Therefore, it is unclear how these algorithms behave on rather small graphs like molecular data. To our knowledge, no subgraph isomorphism comparison directly addresses the problem of searching chemical substructures in molecules. One of the main reasons why such a

*Correspondence: rarey@zbh.uni-hamburg.de
Center for Bioinformatics, University of Hamburg, Bundestraße 43, 20146 Hamburg, Germany

benchmark was not performed in the past was the lack of suitable and publicly available benchmark data sets.

This article describes such various data sets and discusses the differences between the Ullmann and the VF2 subgraph isomorphism algorithm applied on substructures and molecules. In the following, we introduce the graph theoretical concepts, summarize the two algorithms of interest, introduce different benchmark data sets and compare the algorithms' performance in various molecular modeling scenarios.

Preliminaries

For almost 150 years, chemists have used chemical and structural formulas to represent molecules. A structural formula is closely related to the mathematical concepts of graphs which makes graph theory and algorithms directly applicable in cheminformatics.

Graph theoretical background

A *graph* $G = (V, E)$ is defined by a set of nodes V and a set of connecting edges E. The edges of an *undirected* graph have no fixed orientation and if labels are assigned to nodes or edges the graph is denoted as *labeled*. If a path from each node to every other nodes exists, the graph is called *connected*. In the following, all graphs are labeled, undirected and connected except when stated otherwise.

Subgraph isomorphism

Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are *isomorphic* if a bijective projection between nodes V_1 and nodes V_2 exists such that two nodes from V_1 are connected by an edge from E_1 if and only if their image nodes in V_2 are connected by an edge from E_2. An *induced subgraph* of a graph $G = (V, E)$ is defined as a graph $G' = (V', E')$ whose nodes V' are a subset of V and whose edges E' are all possible edges from E that connect two nodes in V'. An *induced subgraph isomorphism* between a query graph G_1 and a target graph G_2 exists if G_1 is isomorphic to an induced subgraph of G_2, i.e., the query graph G_1 is a subgraph of the target graph G_2.

The problem of finding an isomorphic induced subgraph is believed to be a problem for which no efficient solution exists, i.e., it belongs to the class of NP-complete problems [5,24]. Therefore, every subgraph isomorphism algorithm will show exponential run times with respect to the input graph size.

Molecular graphs

A *molecular graph* is given by nodes and edges that represent atoms and bonds, respectively. Often nodes and edges are labeled with atom and bond properties. Obviously, molecular graphs are undirected. The number of edges connecting each node is limited by the number of covalent bonds an atom can form. Therefore, the number of edges in a molecular graph linearly depends on the number of nodes.

Molecules are equal or isomorphic if their molecular graphs are isomorphic and the labels of the atoms and bonds are equal to the labels of their mapped atoms and bonds respectively. When two molecules differ in size, one can be a substructure of the other, i.e., a subgraph isomorphism between the two molecules exists. The small number of atoms and the linear atom degree allow for a fast subgraph isomorphism test on molecules.

Substructure graphs

A *substructure graph* can be a molecule fragment, e.g., a functional group, or a more generalized construct. For example, a single halogen node might represent a fluorine, chlorine, bromine or iodine atom. The same applies to edges, e.g., an edge is either a single or a double bond. In the following, we will use substructure graphs with such general labels. Figure 1 shows an example.

Substructure graphs are compared with molecules to detect subgraph isomorphisms. The goal is to determine the presence or location of a functional group or a specific molecular structure. Nodes and edges are mapped to atoms and bonds in accordance with their labels. Since edges are explicitly assigned to bonds, the detected isomorphic subgraph might not be induced, i.e., non-circular substructures can be mapped to circular molecule parts.

For a clear differentiation, we will use the terms atoms and bonds for molecular target graphs and nodes and edges for query substructure graphs.

Substructure pattern languages

A substructure graph can be formulated by using a substructure pattern language like SMILES Arbitrary Target Specification (SMARTS) [25], Sybyl Line Notation (SLN) [26] or Wiswesser Line Notation (WLN) [27]. All languages define a substructure graph in a textual line notation similar to a molecule's chemical formula. They allow the definition of a substructure's topology and node and bond properties, including logical alternatives. SMARTS even provides the opportunity to specify additional information like a chemical environment. In this study, all substructures are formulated as SMARTS expressions.

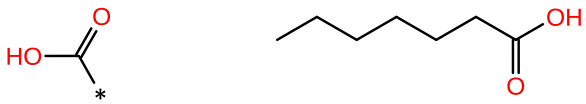


Figure 1 Carboxylic acid pattern and heptonic acid. A carboxylic acid pattern (left) with '*' indicating any atom. Heptonic acid (right).

Methods

The Ullmann and the VF2 algorithms are two algorithms that solve the subgraph isomorphism problem. Applied to substructure and molecular graphs, they can be used to detect substructures in molecules. Both algorithms calculate an exact solution, i.e., the exact substructure must be present, and their application is not restricted to a special class of graphs, i.e., is not limited to molecular graphs.

Ullmann algorithm

The Ullmann algorithm [7] is a backtracking procedure that employs a relaxation-based refinement step to reduce the search space. It operates on a $n \times m$ matrix M of boolean values, where n is the number of substructure nodes and m the number of molecule atoms. An entry at position (i, j) marks the compatibility of labels for substructure node i and molecule atom j. Additionally, it uses a boolean vector f of length m marking mapped atoms. Algorithms 1 and 2 show Ullmann's match and refinement procedure. Figure 2 illustrates one step of the algorithm.

The refinement is the crucial step of the algorithm. It evaluates the surrounding of every possible node-atom mapping. For a valid mapping, every neighbor node must have a compatible atom as illustrated in Figure 3. Otherwise, the mapping is invalid which is marked by setting the corresponding matrix entry to zero. The evaluation takes place for every possible mapping downstream the current row and is repeated until all remaining mappings are valid.

Although the refinement procedure is the key for an efficient reduction of the search space it does not take full advantage of topological constraints. For example, in the case of a small substructure and a large molecule, it evaluates entries topologically too far away from already mapped node-atom pairs.

VF2 Algorithm

The VF2 algorithm [12] iteratively extends a partial solution using a set of feasibility criteria to decide whether to extend or backtrack. It operates on an intermediate algorithm state s which is composed of a partial solution $M(s)$ and adjacency sets $T_1(s)$ and $T_2(s)$. A pair $(n, m) \in M(s)$ represents an atom-node mapping of the partial solution. $M_1(s)$ and $M_2(s)$ describe the atoms and nodes, respectively, that belong to the partial solution. $T_1(s)$ and $T_2(s)$ hold atoms and nodes adjacent to atoms in $M_1(s)$ and nodes in $M_2(2)$, respectively. The algorithm modifies the state s in two steps. From the sets $T_1(s)$ and $T_2(s)$, it creates a candidate set $P(s)$ of atom-node pairs with compatible labels. Then, it explores every candidate $(n, m) \in P(s)$ that fulfills the feasibility rules F_{syn} or backtracks if $P(s)$ is empty. Figure 4 graphically depicts one step of the algorithm.

$F_{syn}(s, n, m)$ (Equation 1) describes the feasibility of candidates (n, m) in state s. It is composed out of two terms, R_{adj} (Equation 2) and R_{inout} (Equation 3). The first feasibility rule R_{adj} guarantees that each atom n' and node m' adjacent (*Adj*) to the atom n and node m of a candidate pair (n, m) are mapped to each other in the partial solution $(n', m') \in M(s)$. The second rule R_{inout} performs a 1-look-ahead in the search procedure based on the nodes' cardinality (*Card*) and allows an early pruning

Algorithm 1

```
1:  input compatibility matrix M, row index k, mapped atoms vector f
2:  output permutation matrix that represents a one-to-one mapping of nodes to atoms
3:  procedure MATCH(M, k, f))                          ▷ first call: Match(M,−1; f = 0)
4:      if k = n then                                  ▷ complete mapping of substructure to molecule
5:          return M
6:      else
7:          for l ← 0 to m − 1 do                      ▷ go over the complete row
8:              if M_(k+1,l) = 1 and f_l = 0 then      ▷ look for a possible mapping
9:                  M_save ← M                         ▷ save state for backtracking
10:             f_save ← f
11:             for j = 0 to m − 1 do
12:                 M_(k+1,j) ← 0                      ▷ remove all possible mappings of current node
13:             M_(k+1,l) ← 1                          ▷ fix one node-atom mapping
14:             f_l ← 1                                ▷ mark atom as used
15:             if Refine(M, k + 1) then               ▷ refine rest of the matrix
16:                 Match(M, k + 1, f)                 ▷ continue with the next row
17:             M ← M_save                             ▷ restore previous solution for backtracking
18:             f ← f_save
```

Algorithm 2

```
1:  input reference to compatibility matrix M, row index k
2:  output true when all substructure nodes still have an option to be mapped onto the molecule
3:  procedure REFINE(&M,k)
4:      repeat
5:          changed ← false                                   ▷ flag that marks matrix changes
6:          for all M(i,j) with i > k and M(i,j) = 1 do       ▷ check all possible mappings (i,j)
7:              valid ← true                                  ▷ flag that marks the valid mapping of a neighbor node
8:              for all x adjacent to i ∈ G1 do               ▷ check all nodes adjacent to node i in substructure G1
9:                  found ← false
10:                 for all y adjacent to j ∈ G2 do           ▷ check all atoms adjacent to atom j in molecule G2
11:                     if M(x,y) = 1 then                    ▷ possible mapping of compatible pair (x,y)
12:                         if edge [i,x] = edge [j,y] then   ▷ edge type is compatible to bond type
13:                             found ← true                  ▷ valid mapping of neighbor found
14:                             break                         ▷ leave loop over adjacent atoms
15:                 if found = false then                     ▷ adjacent node has no possible mapping
16:                     valid ← false
17:                     break                                 ▷ leave loop over adjacent nodes
18:             if valid = false then                         ▷ at least one adjacent node can not be mapped
19:                 M(i,j) ← 0                                ▷ mark mapping of node i to atom j invalid
20:                 changed ← true                            ▷ mark matrix as changed
21:                 if M(i,h) = 0 for 0 ≤ h ≤ m − 1 then      ▷ check if node i can not be mapped anymore
22:                     return false                          ▷ induce backtracking in match procedure
23:     until changed = false                                 ▷ repeat refinement because mapping(s) became invalid
24:     return true
```

of the search tree. Figure 5 and Figure 6 give an illustration of the feasibility rules.

The problem of reaching the same state, i.e., the same partial solution $M(s)$, via different paths is handled by imposing an arbitrary total order $\prec$ onto the subgraph nodes and processing only smallest feasible candidates with regard to that order. Therefore, feasible candidates (n_i, m_j) in $P(s)$ are not processed if $m_k \prec m_j \in P(s)$.

The main difference between the two algorithms is the way they account for the topology of the substructure. The Ullmann algorithm processes a compatibility matrix top-down. In every step it fixes one node-atom mapping and checks all other possible assignments for validity. Therefore, it processes substructure nodes in an non-topological, arbitrary order. In contrast, the VF2 iteratively adds node-atom pairs to a current

$$F_{syn}(s,n,m) = R_{adj} \wedge R_{inout} \tag{1}$$

$$R_{adj} = (\forall n' \in M_1(s) \cap Adj(G_1,n))\exists m' \in Adj(G_2,m)|(n',m') \in M(s)) \wedge (\forall m' \in M_2(s) \cap Adj(G_2,m))\exists n' \in Adj(G_1,n)|(n',m') \in M(s)) \tag{2}$$

$$R_{inout} = Card(Adj(G_1,n) \cap T_1(s)) \geq Card(Adj(G_2,m) \cap T_2(s)) \tag{3}$$

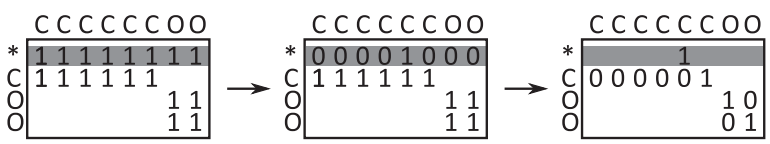


Figure 2 Iteration of Ullmann algorithm. One step of the Ullmann algorithm. The initial compatibility matrix (left) shows carboxylic acid substructure nodes as rows and heptonic acid molecule atoms as columns. A non-zero entry indicates the compatibility of a node-atom pair. Zero entries are not shown. In the current row, indicated in gray, the algorithm choses one compatible node-atom mapping (middle) and refines all unprocessed rows (right). The algorithm continues with the next row. Figure 3 illustrates the refinement.

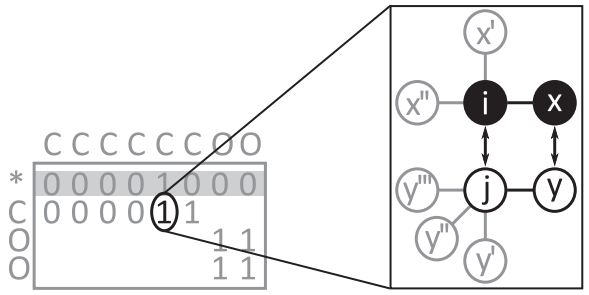


Figure 3 Refinement of Ullmann algorithm. Ullmann refinement step. For a mapping of node *i* to atom *j*, all adjacent nodes *x* must have at least one valid mapping *y*. If this condition is not fulfilled, the mapping (i, j) is invalid and the corresponding matrix entry at (i, j) is set to zero.

solution and therefore directly explores the substructure's topology.

Substructure pattern formulation for efficient computation

The formulation of substructure patterns is a tedious task. Most pattern languages are difficult to read and even more difficult to write, especially when defining isomeric or tautomeric structures. As a result, substructure formulations are focused on a correct chemical representation of a pattern. That formulation might be suboptimal for computational processing. Therefore, we present simple guidelines to optimize patterns for the search in molecules.

For an optimal formulation, the substructure must be in an order that allows an early processing of unusual nodes and edges, rare fragments and functional groups. Obviously, certain elements are more common than others. The same applies for substructure nodes that define a high number of atom properties or are part of an aromatic system. Unusual edges define aromatic bonds or those with a high bond order. Therefore, we write optimized substructures such that nodes with the rarest element, highest property specification and aromaticity as well as high order or aromatic bond definitions occur first. Additionally, we place substructure parts that are rather common or difficult to process at the end of the formulation. Nodes that specify generic atoms, hydrogen atoms, carbon atoms, and ring atoms are common. Chemical environments are difficult to process for most search algorithms, since they enforce an additional search step.

In the following we perform every pattern reformulation by hand. Nevertheless, both algorithms are well suited for an automated optimization process. Ullmann's algorithm processes substructure nodes according to their row numbers in the compatibility matrix. Since row numbers are assigned arbitrarily, they can resemble the order employed by applying the given optimization rules. The VF2 uses an arbitrary node relation to obtain a total order. Therefore, the optimized order can be directly used.

Data sets

Both algorithms are tested in different application setups like complete database scans, substructure-based filter scenarios and individual substructure-molecule searches. The tests show the dependency of the algorithm run times on substructure formulation, substructure size and molecule size.

The data sets comprise 1336 SMARTS from the literature [28-37] and molecules out of ZINC lead-like and ZINC everything database [1]. All data sets are provided in Additional file 1.

Substructure search set

Molecule size is a crucial factor with respect to the algorithmic search time. To explore the influence of molecule size, we select a subset from the initial 1336

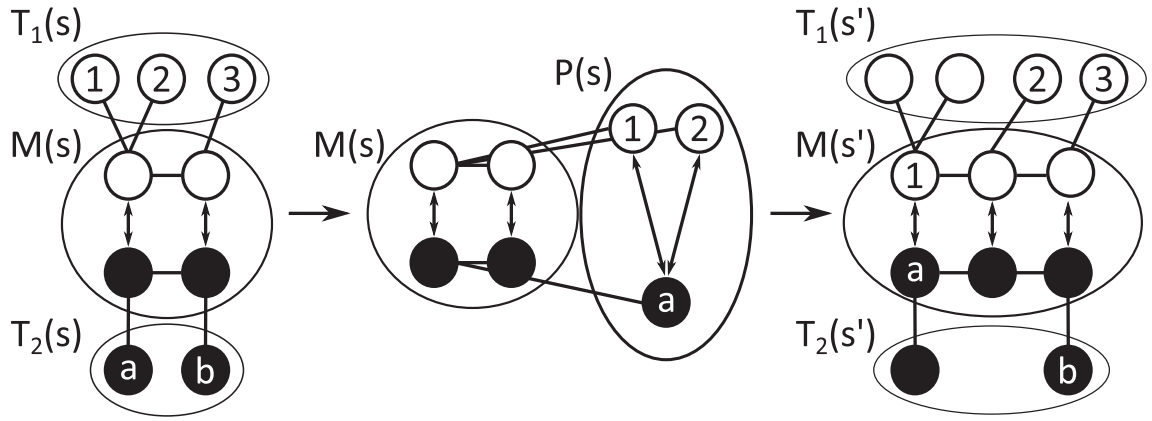


Figure 4 Iteration of VF2 algorithm. One VF2 iteration. The algorithm extends the current solution $M(s)$ of state s by one candidate $(1, a)$ chosen from $P(s)$. $T_1(s)$ and $T_2(s)$ show the nodes adjacent to mapped atoms and nodes.

Algorithm 3

input intermediate state s; first call with $M(s_0) = (n, m)$
output mapping M of substructure $G_2 = (N_2, B_2)$ onto molecule $G_1 = (N_1, B_1)$
procedure MATCH(s)
  if $| M(s) |=| G_2 |$ **then** ▷ all substructure nodes are mapped onto the molecule
    return $M(s)$
  else
    $m \leftarrow m_i \mid \forall m_j \in T_2(s) \backslash \{m_i\} : m_i \prec m_j$ ▷ choose smallest node m according to relation '$\prec$'
    for all $n \in T_1(s)$ **do** ▷ process all ordered atoms adjacent to $M_1(s)$
      if m and n are compatible **then** ▷ check if node-atom labels agree
        $P(s) \cup (n, m)$ ▷ P stores compatible pairs adjacent to M
    for all $p = (n, m) \in P(s)$ **do** ▷ process all possible extensions
      if $F_{syn}(s, n, m)$ **then** ▷ check if node-atom pair is feasible
        $s_{save} \leftarrow s$ ▷ save state for backtracking
        $M(s) \leftarrow M(s) \cup p$ ▷ extend partial solution by one node-atom mapping
        $T_1(s) \leftarrow \bigcup_{n \in M_1(s)} Adj(G_1, n) \setminus M_1(s)$ ▷ update atoms adjacent to the partial solution
        $T_2(s) \leftarrow \bigcup_{m \in M_2(s)} Adj(G_2, m) \setminus M_2(s)$ ▷ update nodes adjacent to the partial solution
        Match(s) ▷ continue with the next extension
        $s \leftarrow s_{save}$ ▷ backtrack

SMARTS. All duplicate expressions, expressions with errors, extensions and those that define isotopes or are disconnected are removed. The resulting set comprises 1235 SMARTS whose property overview is given in the Additional file 2: Table S1. SMARTS allows the explicit formulation of hydrogen atoms and the definition of atom environments. When explicit hydrogen atoms are used a search procedure must evaluate all hydrogen atoms, which roughly doubles the number of atoms to be evaluated. Atom environments induce an additional search step during the actual search procedure. In order to circumvent misinterpretations of the results, we group the SMARTS patterns by the presence/absence of explicit hydrogens and recursive environments into individual sets. The Additional file 2: Table S2 – S19 give detailed statistics on SMARTS properties for every set.

The final sets contain all SMARTS patterns for which 100 molecules containing the pattern could be randomly selected from ZINC lead-like and ZINC everything.

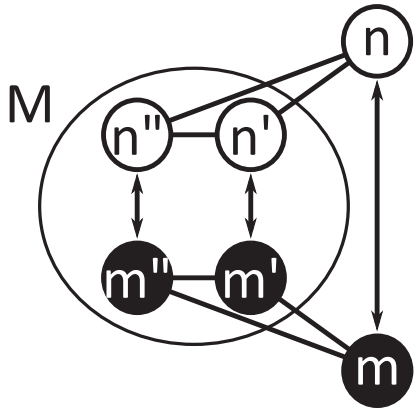


Figure 5 VF2 feasibility rule for node cardinality. VF2 feasibility rule for node cardinality. The rule guaranties a one-to-one mapping of edges in the current solution $M(s)$. For a candidate mapping (n, m), all atoms (n' and n'' in $M_1(s)$) adjacent to n must be mapped to the corresponding nodes (m' and m'' in $M_1(s)$) adjacent to m. Otherwise the candidate mapping is not feasible.

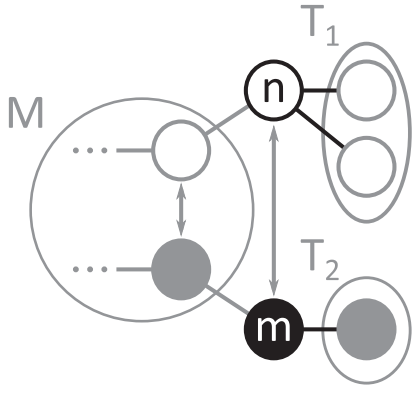


Figure 6 VF2 feasibility rule for node cardinality (1-look-ahead). VF2 feasibility rule for node cardinality (1-look-ahead). The rule prohibits an extension of the current solution $M(s)$ by candidates with a substructure cardinality that can not be fully mapped onto the graph. In the given example, node m has one edge into $T_2(s)$ and is mapped to atom n with a cardinality of two. Therefore, the mapping is feasible.

Table 1 shows the number of SMARTS for which the selection process was successful. The molecular property distribution of each set is similar to the corresponding ZINC database as shown in the Additional file 2: Table S23 – S24.

Molecule search set

Substructure size is the second major factor regarding pattern matching time. A set to measure its impact is composed by randomly selecting molecules from ZINC lead-like containing all-in-all 80 different substructures of various size. The presence of so many substructures in a single molecule is rather rare but selecting molecules with less patterns gives poor results. A selection was only possible for the set of SMARTS having no explicit hydrogen nodes and no recursive environments. The other three sets contain patterns of much higher complexity which are rarely present in one molecule or patterns that are designed to be complementary to each other, e.g., PAINS.

PAINS substructure set

For a detailed case study, we choose 16 PanAssayINterferenceStructures(PAINS) described by Baell et al. [38] as 'filter family A'. The PAINS substructures should describe unspecific binders in protein-protein interaction assays. PAINS were originally given in SLN and converted to SMARTS by Rajarshi Guha using Cactvs [39]. The converted PAINS patterns include hydrogen atoms and recursive environments. The PAINS's property distribution is shown in the Additional file 2: Table S20 and Additional file 3: Figure S2 – S5 depict each substructure.

Worst-case test

Since highly symmetric molecules impose a challenge for substructure search algorithms, we test a phenylring query against a fulleren target as a worst-case search scenario.

Database subset

The database subset comprises the first 100.000 molecules from ZINC lead-like as of February 12th, 2011 and is designed to resemble a complete database. Its property distribution is similar to that of the full ZINC lead-like database as shown in Additional file 2: Table S25.

Results and discussion

Search speed is measured on a single Intel(R) Xeon(R) CPU E5630 2.53GHz core. Each matching is repeated 400 times and average values are recorded. Average matching times are raw matching times excluding File I/O, molecule initialization and post-processing of search results, i.e., matching time only.

We are aware of the fact that the evaluation is done with an example implementation of both algorithms that most likely has some room for optimization. Nevertheless, we believe that our results allow general conclusions regarding the algorithms' behavior on molecular data.

Overall search speed

An overview of the VF2 and Ullmann matching times is shown in Figure 7. The times are measured on the 46900 substructure-molecule pairs of the Substructure Search Set. Both substructure algorithms search for all occurrences of each substructure. The histograms show that both algorithms have most match times in a range below 1 milliseconds (ms) (92.3% VF2, 73,4% Ullmann) with a median of 0.04ms for the VF2 and 0.1ms for the Ullmann, respectively. While the maximum VF2 matching time is below 30ms, the Ullmann shows times of more than 100ms for 1.12% (5352 pairs) and more than 1000ms for 0.22% (104 pairs) of the data set. Interestingly, the Ullmann search times do not change drastically in the case where the search is constrained to the first occurrence of each substructure. In contrast, the VF2 outlier times drop down by one half. In conclusion, both algorithms can solve most instances in reasonable time and the median run times differ by a factor of 2.5 betwenn VF2 and Ullmann's algorithm. In rare cases, the Ullmann algorithm is up to 1000 times slower than the VF2.

Explicit vs. implicit hydrogens

A closer analysis of Ullmann and VF2 matching times reveals a slight increase in run times for SMARTS patterns with explicit hydrogens, which is documented by the histograms in Figure 8. The median search times of the VF2 are 0.08ms for substructures with only implicit hydrogens and 0.19ms with explicit hydrogens, 0.22ms and 1.09ms for the Ullmann, respectively. In accordance, the maximum run time of the VF2 doubles, while that of the

Table 1 SMARTS, ZINC lead-like, ZINC everything test sets

	all SMARTS		ZINC lead-like set		ZINC everything set	
	no H nodes	H nodes	no H nodes	H nodes	no H nodes	H nodes
no recursion	504	432	347	56	400	43
recursion	234	65	48	18	106	39

All processable SMARTS split by the presence/absence of explicit hydrogen nodes and recursive environment specifications and the subsets used in the ZINC lead-like and ZINC everything set.

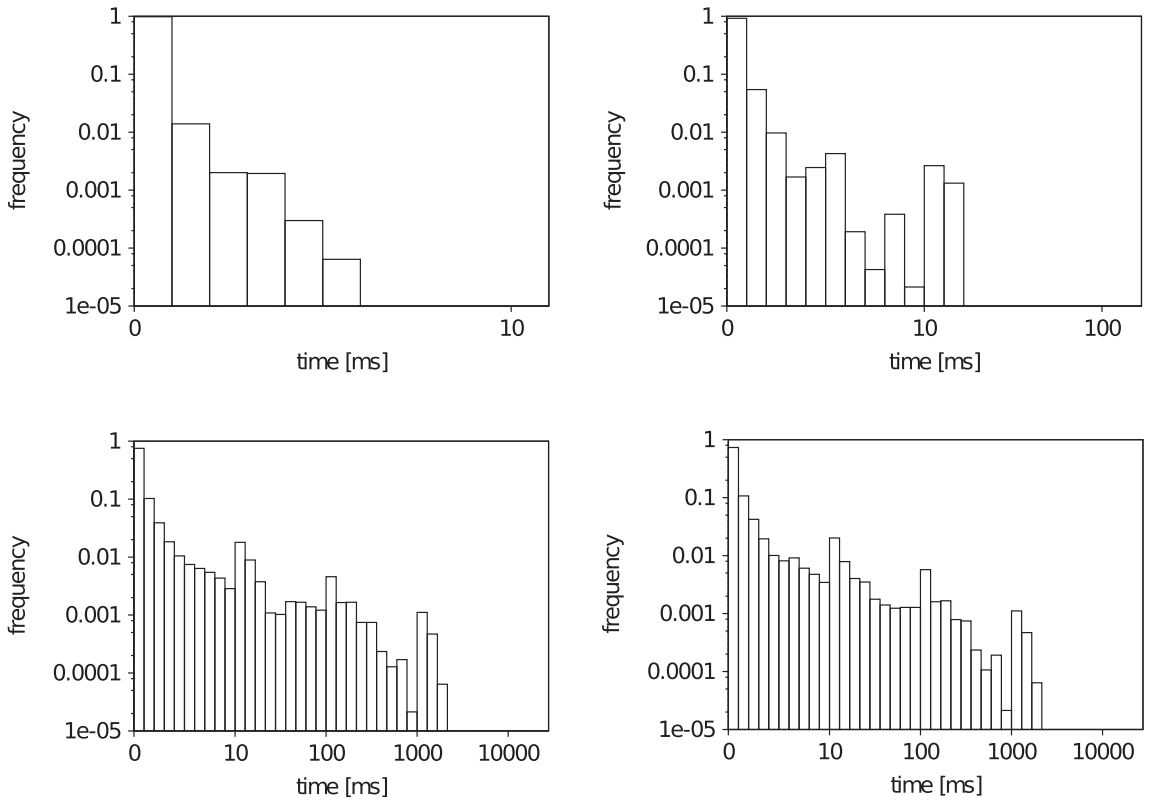


Figure 7 Overall run time histogram. Histogram over VF2 (top) and Ullmann (bottom) matching times on the Substructure Search Set. The algorithms search for the first (left) and all (right) occurrence(s) of the substructure. All plots are double logarithmic and times are given in milliseconds (ms).

Ullmann algorithm is roughly four times larger. The reason for an increase in run times is twofold. About 50% of atoms in a small molecule are hydrogens. Therefore, when matching patterns with explicit hydrogens, in contrast to patterns with only implicit hydrogens, all hydrogen atoms have to be evaluated. This doubles the number of evaluated atoms during the search, and hence, increases the run time. Additionally, for every hydrogen node, an explicit placement must be found, as opposed to the comparison of the sheer number of hydrogens attached to an atom. This raises the number of evaluated atoms as well as the number of found mappings, and therefore increases the run time.

Recursion vs. no recursion

An interesting aspect of the SMARTS pattern language is the ability to recursively define the chemical environment of an atom. To match a pattern that includes one or more nodes with atom environments, a subgraph search algorithm has to recursively perform a subgraph isomorphism test during the actual search. Figure 9 shows the impact on matching times when recursive environments are defined. Median run times for the VF2 are 0.04 ms for SMARTS without and 0.35 ms for SMARTS with environment specifications, 0.15 ms and 4.87 ms for Ullmann's algorithm, respectively. Surprisingly, the Ullman algorithm is much more sensitive to recursive patterns. The presence of environment specifications can lead to a 30 times increase in Ullmann matching times while VF2 times maximal rise by a factor of two. The sensitivity is due to the fact that Ullmann's algorithm creates a matrix that represents all possible mappings of nodes to atoms. Since most recursive environments are rather small, the construction and evaluation of such a matrix represents a computational overhead that is reflected in an increase of the overall search time.

Molecule size

In order to explore the influence of molecule size we examine 469 substructure-molecule pairs from the Substructure Search Set. As almost all results are similar, we chose only some representative substructure-molecule pairs shown in Figure 10. All figures, given in Additional file 1, show a significantly smaller matching time for the VF2 and a linear influence of the molecule size on the matching time. The difference between VF2 and Ullmann

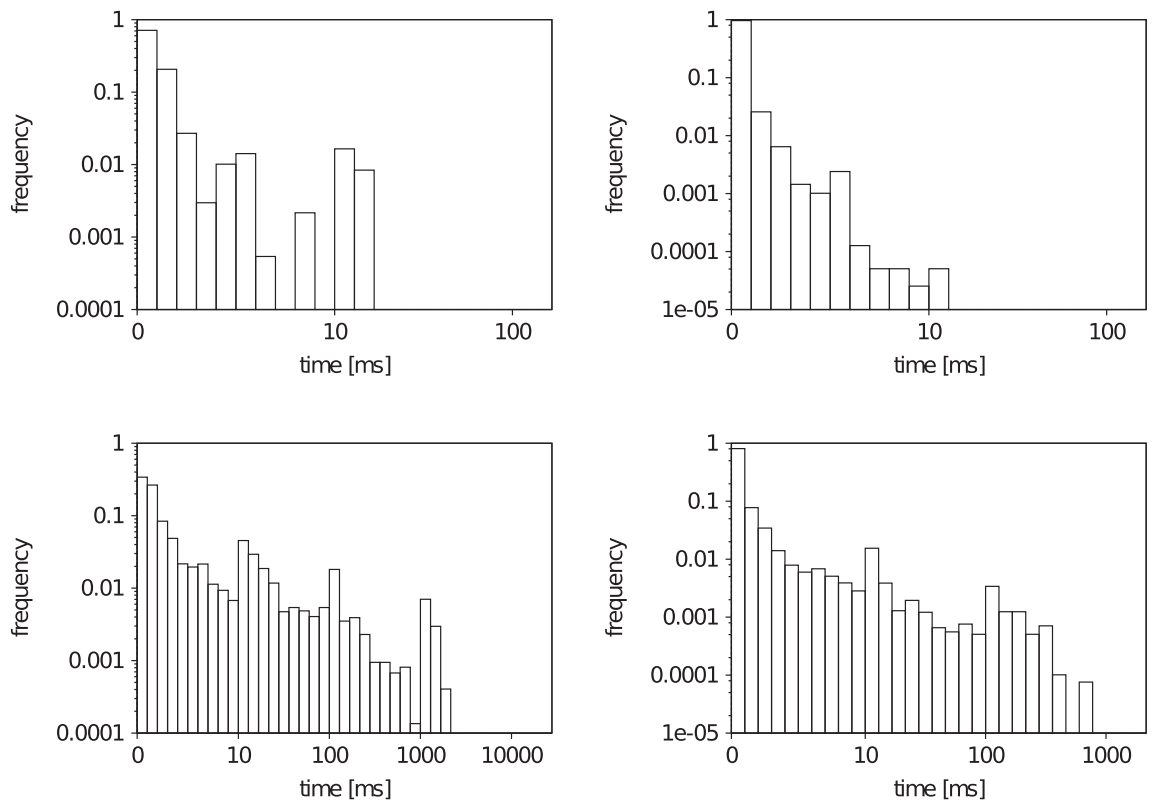


Figure 8 Explicit vs. Implicit hydrogens run time histogram. Histogram of VF2 (up) and Ullmann (down) matching times with (left) and without (right) explicit hydrogens on the Substructure Search Set. The algorithms search for all occurrences of the substructure. All plots are double logarithmic and times are given in milliseconds (ms).

matching times becomes even more prominent when examining the cases where explicit hydrogens (Figure 11 top-left), recursive environments (Figure 11 bottom-right) or both (Figure 11 bottom-left) are present. The linear impact of the molecule size on the run time is explained by the constant number of bonds an atom can form as can be obtained from a theoretical analysis of backtracking algorithms for subgraph isomorphism [40,41].

Subgraph size

The impact of subgraph size regarding the matching time was evaluated with a meaningful test set for substructures with only implicit hydrogens and no recursive environments. Unfortunately, a suitable test set could only be constructed for SMARTS patterns without explicit hydrogens and recursive environments. From observing 100 molecules in which at least 80 substructures with different size could be matched, we assume an exponential run time development with increasing subgraph size for both algorithms. The exponential increase seems to be slower for the VF2 in all cases. An example is given in Figure 12 and all plots are provided in Additional file 1. The difference in matching times drastically decreases when only the presence of a substructure, rather than all occurrences, is of interests. The exponential match time of both algorithms regarding the substructure size is again in agreement with a theoretical analysis of the subgraph isomorphism problem [40,41].

Worse-case test

As a worse-case substructure search scenario, we test a phenyl-ring query against a C70 fullerene target. The Ullmann finds the first occurrence in 51.11 ms and all matches in 106.94 ms. The VF2 is about 130 times faster when it solves the problem for the first occurrence (0.39 ms) and about 5 times when searching for all matches (21.67ms). Clearly, the phenyl-fullerene example is not the worse-case when considering SMARTS substructures. Substructures with explicit hydrogen nodes or recursive atom environments yield much higher run times. Nevertheless, the phenyl-fullerene experiment gives good guidance on how the Ullmann and VF2 algorithms behave on highly symmetrical structures.

Complete database search

Often substructure search algorithms are used in database search scenarios in which a database is scanned for all

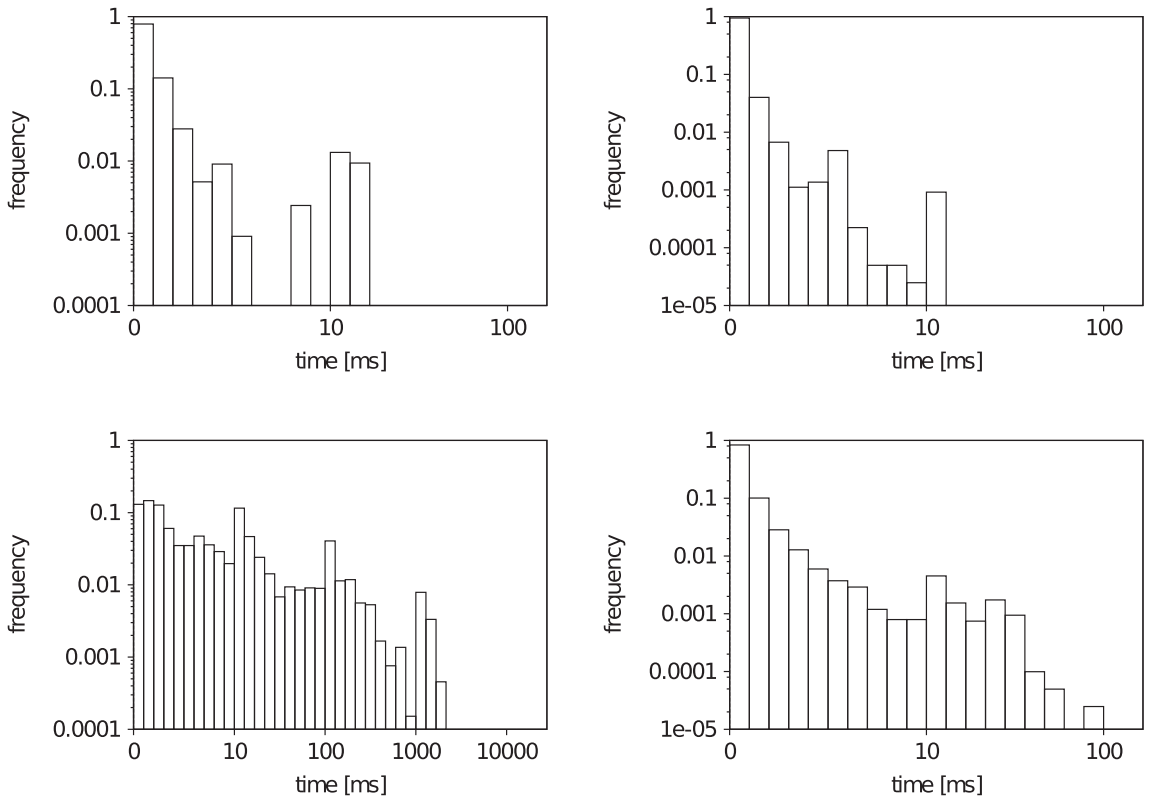


Figure 9 Recursion vs. no Recursion run time histogram. Histogram of VF2 (up) and Ullmann (down) matching times with (left) and without (right) recursive environments on the Substructure Search Set. The algorithms search for all occurrences of the substructure. All plots are double logarithmic and times are given in milliseconds (ms).

molecules that contain a given query structure. Even though most database search systems are able to eliminate a large number of molecules from the actual subgraph isomorphism search using screening techniques [10,22,41,43-46], a remarkable number of molecules might remain. The following test simulates a sequential subgraph isomorphism test over a large set of molecules. We search all 1235 patterns from the Substructure Search Set against the Database Subset and measure the complete time to identify all molecules which contain such a substructure. Since the majority of the first 100.000 molecules of the ZINC lead-like database do not contain a given pattern, the search time is dominated by the algorithm's ability to quickly identify the non-occurrence of a substructure in a molecule. A good screening method would of course enrich the molecules submitted to the isomorphism test with molecules containing the substructure of interests. Nevertheless, testing both algorithms for the ability of quickly detecting molecules without a given pattern will reveal further insights into the algorithmic behavior. This test is only performed once, as minor changes in run time do not affect the order of magnitude.

From the two histograms in Figure 13, it is clear that the VF2 algorithm is much faster in sequentially scanning a large number of molecules. The median search time of the VF2 is 2.84 s and 38.7 s for the Ullmann. The VF2 algorithm finishes 53.06% of the search queries below 10s and 97.61% below 10^2 s, while the Ullmann completes 3.73% below 10s, 54.24% below 10^2 s, 92.36% below 10^3 s (16.6 min), 98.76% below 10^4 s (2.78 h) and 99.85% below 10^5s (27.78 h). All in all, in rare instances a database search system that uses the Ullmann algorithm might need over a day to give results for a single query, even though, most of the molecules might be eliminated from the subgraph isomorphism test.

Parallelization scaling

The subgraph isomorphism problem is nearly perfectly suited for parallel computing when matching one query structure against many target structures. One simple but effective solution is a parallelization by data separation of the target structures. An alternative is an algorithm level parallelization based on the algorithms' recursion. Since most substructure searches are below 1ms and most molecules consist of less than 100 atoms, a parallelization of one substructure against one target search is most likely not as efficient as searching in parallel on the data

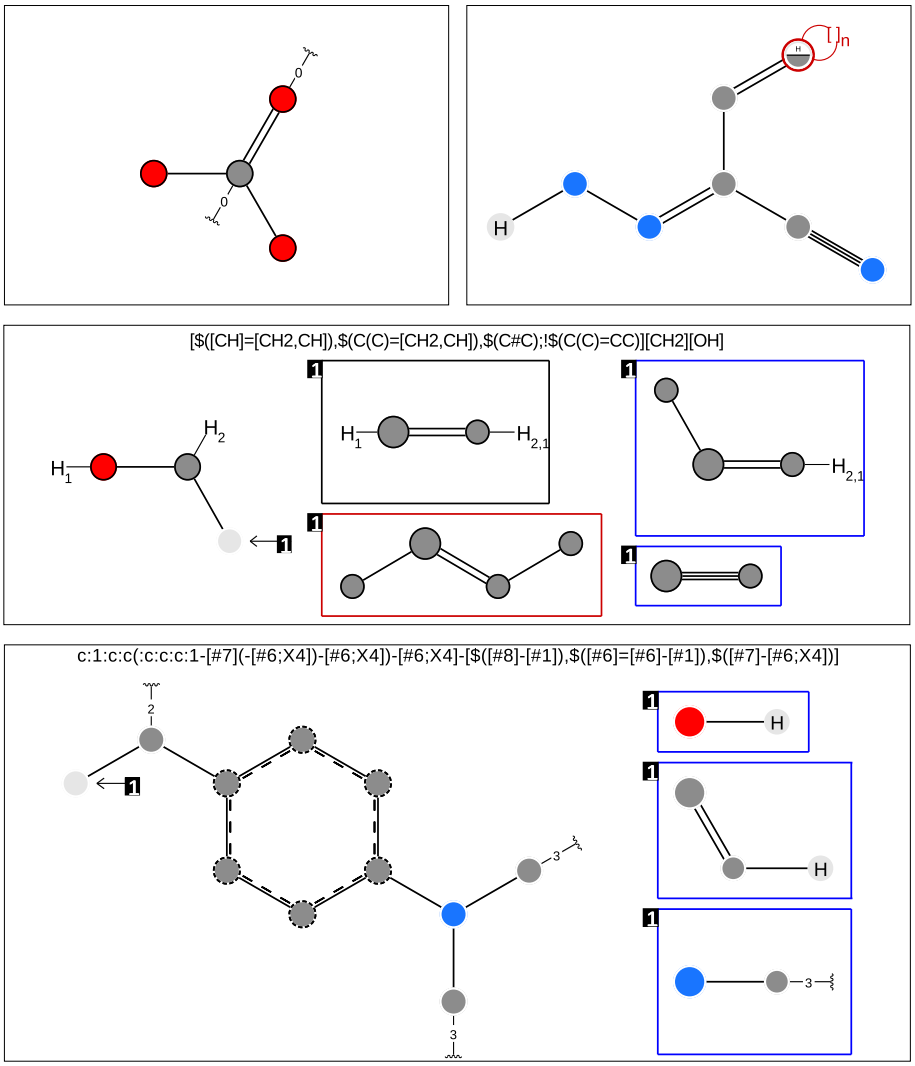


Figure 10 Molecule size experiment patterns. Depiction of SMARTS pattern with no explicit hydrogens and no recursion (top-left), explicit hydrogens and no recursion (top-right), no explicit hydrogens and recursion (middle) and with explicit hydrogens and recursive atom environments (bottom). The legend can be found in the Additional file 3: Figure S1. Depictions are created by SMARTSViewer [42].

level. The situation might change when searching large query substructures against large target structures, e.g., searching for motifs in proteins.

In order to evaluate the efficiency of data level parallelization, we test both algorithms with the same data separation strategy on the PAINS Substructure Set against the complete ZINC lead-like database on different numbers of CPU cores. The target structures are split into equal blocks such that each core gets the query structures and a the same number of molecules. The measurement on one core is performed in sequential and parallel mode so that the computational overhead for parallelization becomes directly present. Detailed tables on the matching times and scaling factors on different numbers of cores can be found in Additional file 2: Table S26 – S27.

Both algorithm show good scaling behavior on all instances. On 8 cores the search times are decreased by an average factor of 5.6 for the VF2, and 6.92 for Ullmann's algorithm respectively. The overall slightly better scaling of the Ullmann algorithm can be explained by the longer matching times. Longer matching times reduce the parallelization overhead relative to the matching time.

SMARTS pattern case studies

To explore the possibility of reducing search speed by rearranging the subgraph formulation we created three different formulations for each substructure of the PAINS Substructure Set. The *original* substructure formulation as created by Cactvs, an *optimized* version by applying

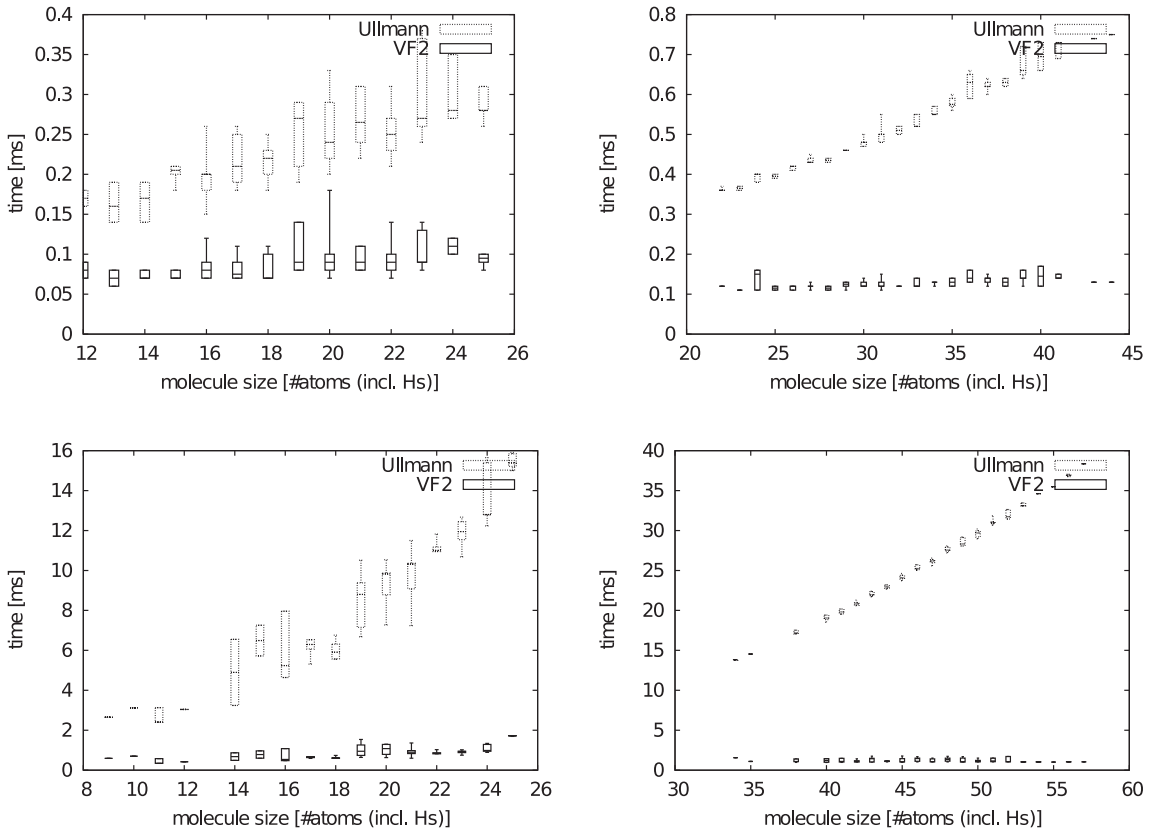


Figure 11 Molecule size search example. Run time comparison between Ullman and VF2 searching for all substructure occurrences with various molecule sizes. The different plots show a linear increase in run time with respect to the molecule size. The top-left pattern does not include explicit hydrogens nor recursive environments. The top-right pattern does include explicit hydrogens but not recursive environments. The bottom-left pattern does not include explicit hydrogens but recursive environments. The bottom-right pattern includes explicit hydrogens and recursive environments. Figure 10 shows a graphical depiction of all four patterns. Times are given in milliseconds (ms).

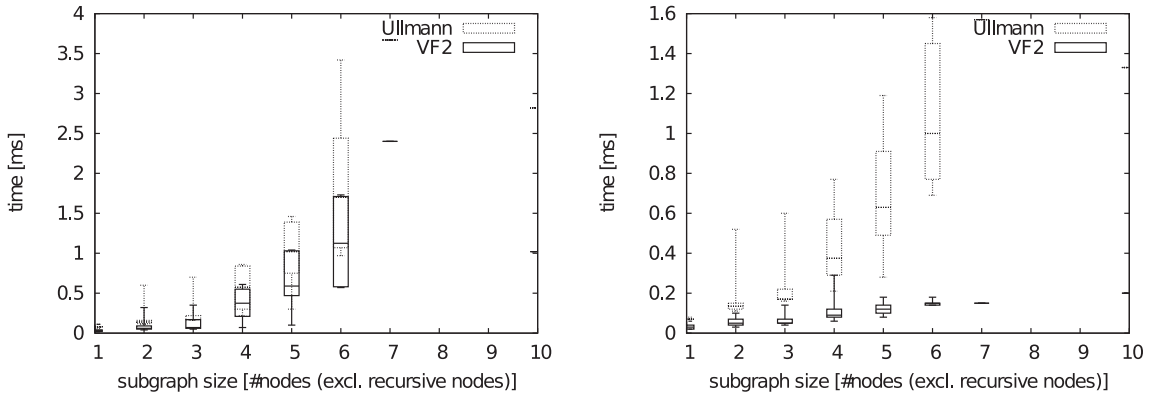


Figure 12 Subgraph size search example. Run time comparison between Ullman and VF2 searching for all (left) and the first (right) substructure occurrence(s) with varying subgraph size. The plots show an exponential increase in run time with respect to the substructure size. Times are given in milliseconds (ms).

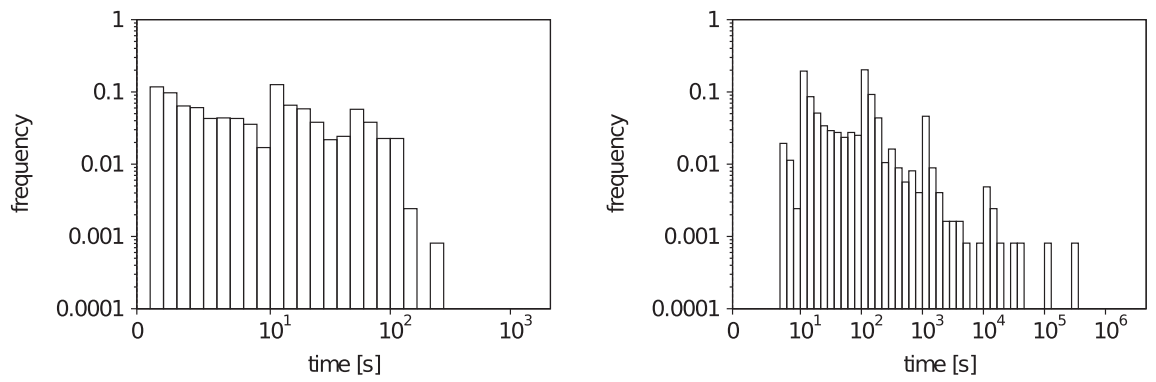


Figure 13 Database scan run time histogram. Run time histogram for the VF2 (left) and Ullmann (right) when searching the first 100.000 molecules from ZINC lead-like for the first substructure occurrences. Both plots are double logarithmic and times are given in seconds(s).

the re-formulation guidelines described in the "Substructure Pattern Formulation" section, and an *anti-optimized* version by applying the rules in reverse. All three formulations are searched against the complete ZINC lead-like database.

As can be observed from the two most extreme cases shown in Table 2, the VF2 algorithm shows run time decreases of up to 13.37 times for the optimized substructure formulations. In accordance, the run time increases up to 15.64 times for the anti-optimized formulation. Surprisingly, the Ullmann algorithm shows no significant change in run time, neither for the optimized nor for the anti-optimized version in all test cases.

Ullman faster than VF2

In almost all test-cases, we see a superior matching performance of VF2 compared to Ullmann's algorithm. In order to exclude the possibility of errors in our time measurements, we re-calculate the benchmarks for all cases in which Ullmann's algorithm shows a smaller matching time than the VF2. The number of repetitions for each search call is increased to 100.000 to increase the time measurement accuracy. Table 3 shows the re-measurement for 10 examples. Clearly, the first measurements were sufficiently accurate and in all these cases the Ullmann outperformed the VF2. To investigated if the subgraph formulation might be unfortunate for the VF2 algorithm, the test is repeated with optimized substructure formulations. The matching times given in Table 3 show that the VF2 is faster in all cases when given an optimized substructure formulation.

Conclusions

We presented, to our knowledge, the first comparison between Ullmann and VF2 subgraph isomorphism algorithm on molecular data and the first data set to perform such a benchmark. Using SMARTS as molecular substructure language, we explored the influence of substructure and molecular size as well as the usage of explicit hydrogen nodes and recursive environment specification on the matching time. Both algorithms where additionally tested for the use in complete database scans and their ability for data-based parallelization. Additionally, we presented an optimization strategy to reduce matching times by substructure pattern reformulation.

In conclusion, the VF2 algorithm outperforms the Ullman in all test cases when supplied with a favorable

Table 2 Optimization run time examples

	Ull. time [s]	Ull. speedup	VF2 time [s]	VF2 speedup	matches
PAINS 4					
original	157139.71	1.00	170.42	1.00	11699
optimized	157027.63	1.00	168.56	1.01	11699
anti-opt.	154195.33	1.02	2664.49	-15.64	11699
PAINS 12 original	3119.04	1.00	1698.42	1.00	9056
optimized	2142.41	1.46	142.28	11.94	9056
anti-opt.	3077.34	1.01	1675.40	1.01	9056

Two examples of searching the PAINS Substructure Set against the complete ZINC lead-like database. Ullmann and VF2 times in seconds and speed ups are shown. Results for all 16 PAINS are given in the Additional file 2: Table S28 – S29 and the re-formulated PAINS in Additional file 2: Table S30.

Table 3 Ullmann faster than VF2 without optimization examples

SMARTS	Ullmann time [ms]	VF2 time [ms]
[#6]C(=[O,SX2])[CX4]C(=[O,SX2])[#6]	0.868	0.948
[O,SX2]=C([#6])[CX4]C(=[O,SX2])[#6]	0.654	0.271
[#6]C(=[O,SX2])C(=[O,SX2])[#6]	0.938	1.046
[O,SX2]=C([#6])C(=[O,SX2])[#6]	0.668	0.203
[a]~*~*-[CH3]	0.479	0.601
[CH3]-*~*~[a]	0.209	0.074
[C](=O)([C,c,O,S])[C,c,O,S]	0.400	0.558
O=[C]([C,c,O,S])[C,c,O,S]	0.403	0.144
[CD3H0,R](=[SD1H0])([ND2H1,R])([ND2H1,R])	0.251	0.510
[SD1H0]=[CD3H0,R]([ND2H1,R])([ND2H1,R])	0.242	0.076
[nD3H0,R](~[OD1H0])(a)a	0.290	0.435
[OD1H0]~[nD3H0,R](a)a	0.290	0.091
[R](-*(-*))~*~*~*~[a]	2.082	2.774
[a]~*~*~*~[R](-*(-*))	1.764	0.906
c([OH])c([OH])c([OH])	0.581	0.708
[OH]cc([OH])c([OH])	0.581	0.274
c1([OH])c(O[CH3])cccc1	0.805	0.947
[OH]c1c(O[CH3])cccc1	0.797	0.169
c1([OH])ccc(O[CH3])cc1	0.74	0.922
[OH]c1ccc(O[CH3])cc1.	0.734	0.193

Examples for SMARTS without explicit hydrogens and recursive environments for which the Ullmann algorithm shows a superior run time compared to the VF2. Time measurements are averages over 100.000 search repetitions in milliseconds. Times are shown for the original SMARTS formulation (top) and an optimized version (bottom) according to our guidelines.

substructure formulation and seems to be more robust in terms of run time outliers. Even though the VF2 is generally faster, both algorithms perform most single substructure-molecule searches in times below one millisecond, which seems acceptable for most cheminformatics applications. Nevertheless, we recommend using the VF2 algorithm for molecular substructure searching in cheminformatics software because it shows a general run time superiority of about one order of magnitude.

The syntactic formulation of a substructure in terms of arrangement might be a critical point for the underlying subgraph isomorphism algorithm. Our experiments show that the VF2 algorithm is sensitive to the substructure's formulation while the Ullmann algorithm is not. Therefore, other subgraph isomorphism algorithms might show the same sensitivity and need to be experimentally tested.

Fortunately, the subgraph reformulation rules as shown here have not to be done by hand. The VF2 algorithm is based on a precalculated node order which can be manipulated following the reformulation rules. Due to the sensitivity of the VF2 algorithm for node rearrangements, the algorithm has further room for optimization.

Additional files

Additional file 1: Additional data (Additional file 1). /datasets/smarts/literature_Hs_noRec.smarts. SMARTS substructures with hydrogens and no recursion.
SMARTS substructure patterns with hydrogens and no recursive atom environments.
/datasets/smarts/literature_Hs_rec.smarts. SMARTS substructures with hydrogens and recursion.
SMARTS substructure patterns with hydrogens and with recursive atom environments.
/datasets/smarts/literature_noHs_noRec.smarts. SMARTS substructures without hydrogens and no recursion.
SMARTS substructure patterns without hydrogens and no recursive atom environments.
/datasets/smarts/literature_noHs_rec.smarts. SMARTS substructures without hydrogens and with recursion.
SMARTS substructure patterns without hydrogens and with recursive atom environments.
/datasets/smarts/pains_p_m150_antioptimized.txt. PAINS substructures anti-optimized. PAINS substructures as SMARTS in anti-optimized formulation.
/datasets/smarts/pains_p_m150_antioptimized.txt. PAINS substructures anti-optimized.
PAINS substructures as SMARTS in anti-optimized formulation.
/datasets/smarts/pains_p_m150_original.txt. PAINS substructures original.
PAINS substructures as SMARTS in original formulation as obtained from the literature.

/datasets/substructure_search_set/literature_Hs_noRec.smarts.everything.benchmarkset. Substructure Search Set, explicit hydrogens and no recursion, ZINC everything
Search set to test the run time influence of the molecule size. Substructures are in SMARTS and do contain explicit hydrogens but no recursive atom environments. For each substructure pattern 100 molecules that contain the pattern were selected at random from ZINC everything. Substructures and molecules are given as space separated SMARTS and SMILES.
/datasets/substructure_search_set/literature_Hs_rec.smarts.everything.benchmarkset. Substructure Search Set, explicit hydrogens and with recursion, ZINC everything.
Search set to test the run time influence of the molecule size. Substructures are in SMARTS and do contain explicit hydrogens and recursive atom environments. For each substructure pattern 100 molecules that contain the pattern were selected at random from ZINC everything. Substructures and molecules are given as space separated SMARTS and SMILES.
/datasets/substructure_search_set/literature_noHs_noRec.smarts.everything.benchmarkset. Substructure Search Set, no explicit hydrogens and no recursion, ZINC everything.
Search set to test the run time influence of the molecule size. Substructures are in SMARTS and do not contain explicit hydrogens or recursive atom environments. For each substructure pattern 100 molecules that contain the pattern were selected at random from ZINC everything. Substructures and molecules are given as space separated SMARTS and SMILES.
/datasets/substructure_search_set/literature_noHs_rec.smarts.everything.benchmarkset. Substructure Search Set, no explicit hydrogens and with recursion, ZINC everything.
Search set to test the run time influence of the molecule size. Substructures are in SMARTS and do not contain explicit hydrogens but recursive atom environments. For each substructure pattern 100 molecules that contain the pattern were selected at random from ZINC everything. Substructures and molecules are given as space separated SMARTS and SMILES.
/datasets/substructure_search_set/literature_Hs_noRec.smarts.lead-like.benchmarkset. Substructure Search Set, explicit hydrogens and no recursion, ZINC lead-like.
Search set to test the run time influence of the molecule size. Substructures are in SMARTS and do contain explicit hydrogens but no recursive atom environments. For each substructure pattern 100 molecules that contain the pattern were selected at random from ZINC lead-like. Substructures and molecules are given as space separated SMARTS and SMILES.
/datasets/substructure_search_set/literature_Hs_rec.smarts.lead-like.benchmarkset. Substructure Search Set, explicit hydrogens and with recursion, ZINC lead-like.
Search set to test the run time influence of the molecule size. Substructures are in SMARTS and do contain explicit hydrogens and recursive atom environments. For each substructure pattern 100 molecules that contain the pattern were selected at random from ZINC lead-like. Substructures and molecules are given as space separated SMARTS and SMILES.
/datasets/substructure_search_set/literature_noHs_noRec.smarts.lead-like.benchmarkset. Substructure Search Set, no explicit hydrogens and no recursion, ZINC lead-like.
Search set to test the run time influence of the molecule size. Substructures are in SMARTS and do not contain explicit hydrogens or recursive atom environments. For each substructure pattern 100 molecules that contain the pattern were selected at random from ZINC lead-like. Substructures and molecules are given as space separated SMARTS and SMILES.
/datasets/substructure_search_set/literature_noHs_rec.smarts.lead-like.benchmarkset. Substructure Search Set, no explicit hydrogens and with recursion, ZINC lead-like.
Search set to test the run time influence of the molecule size. Substructures are in SMARTS and do not contain explicit hydrogens but recursive atom environments. For each substructure pattern 100 molecules that contain the pattern were selected at random from ZINC lead-like. Substructures and molecules are given as space separated SMARTS and SMILES.
/datasets/substructure_search_set/literature_noHs_rec.smarts.lead-like.benchmarkset. Substructure Search Set, no explicit hydrogens and with recursion, ZINC lead-like.
Search set to test the run time influence of the molecule size. Substructures are in SMARTS and do not contain explicit hydrogens but recursive atom environments. For each substructure pattern 100 molecules that contain the pattern were selected at random from ZINC lead-like. Substructures and molecules are given as space separated SMARTS and SMILES.
/datasets/molecule_search_set/literature_noHs_noRec.smarts.everything.80.benchmarkset. Molecule Search Set ZINC everything.
Search set to test the run time influence of the substructure size. Substructures are in SMARTS and do not include explicit hydrogen nodes or recursive atom environments. For each molecule 80 substructures that are contained in the molecule were selected at random from ZINC everything. Molecules and substructures are given as space separated SMILES and SMARTS.
/datasets/worst_case.benchmarkset. Worst Case Set.
A worst-case substructure search scenario of searching for a phenyl-ring in a highly symmetrical fullerene. Substructure and molecule are in SMARTS and SMILES. **/datasets/zinc_lead-like_2011-02-12_first100k.smi. First 100k Molecules of ZINC lead-like.**
The first 100.000 molecules of the ZINC lead-like database. Molecules are in SMILES.
/results/molecule/allPlots.lead-like.all.eps. Molecule Search Experiment ZINC lead-like.
Experiment to test the run time influence of the substructure size. Plots are box plots showing subgraph size vs. run time for Ullmann and VF2. Both algorithms are set to find all occurrences of a substructure. Molecules were chosen at random from ZINC lead-like.
/results/molecule/allPlots.lead-like.first.eps. Molecule Search Experiment ZINC lead-like.
Experiment to test the run time influence of the substructure size. Plots are box plots showing subgraph size vs. run time for Ullmann and VF2. Both algorithms are set to find first occurrences of a substructure. Molecules were chosen at random from ZINC lead-like.
/results/subgraph/allPlots.lead-like.all.eps. Subgraph Search Experiment ZINC lead-like.
Experiment to test the run time influence of the molecule size. Plots are box plots showing molecule size vs. run time for Ullmann and VF2. Both algorithms are set to find all occurrences of a substructure. Molecules were chosen at random from ZINC lead-like.
/results/subgraph/allPlots.lead-like.first.eps. Subgraph Search Experiment ZINC lead-like.
Experiment to test the run time influence of the molecule size. Plots are box plots showing molecule size vs. run time for Ullmann and VF2. Both algorithms are set to find first occurrences of a substructure. Molecules were chosen at random from ZINC lead-like.

Additional file 2: Supplementary Information (Additional file 2).
Table S1. Profile over the number of property occurrences of all 1235 SMARTS sub-structures.
Table S2. Profile over the number of property occurrences of 738 SMARTS substructures without explicit hydrogens.
Table S3. Profile over the number of property occurrences of 497 SMARTS substructures with explicit hydrogens.
Table S4. Profile over the number of property occurrences of 936 SMARTS substructures without recursive atom environments.
Table S5. Profile over the number of property occurrences of 299 SMARTS substructures with recursive atom environments.
Table S6. Profile over the number of property occurrences of 504 SMARTS substructures without hydrogen atoms and without recursion.
Table S7. Profile over the number of property occurrences of 234 SMARTS substructures without hydrogen atoms and with recursion.
Table S8. Profile over the number of property occurrences of 432 SMARTS substructures with hydrogen atoms and without recursion.
Table S9. Profile over the number of property occurrences of 65 SMARTS substructures with hydrogen atom and with recursion.
Table S10. Profile over the number of property occurrences of 469 SMARTS substructures used in ZINC lead-like benchmark set.
Table S11. Profile over the number of property occurrences of 347 SMARTS substructures with no hydrogen atoms and no recursion in ZINC lead-like benchmark set.
Table S12. Profile over the number of property occurrences of 48 SMARTS substructures with no hydrogen atoms and recursion in ZINC lead-like benchmark set.
Table S13. Profile over the number of property occurrences of 56 SMARTS

substructures with hydrogen atoms and no recursion in ZINC lead-like benchmark set.
Table S14. Profile over the number of property occurrences of 18 SMARTS substructures with hydrogen atoms and recursion in ZINC lead-like benchmark set.
Table S15. Profile over the number of property occurrences of 588 SMARTS substructures used in ZINC everything benchmark set.
Table S16. Profile over the number of property occurrences of 400 SMARTS substructures with no hydrogen atoms and no recursion in ZINC everything benchmark set.
Table S17. Profile over the number of property occurrences of 106 SMARTS substructures with no hydrogen atoms and recursion in ZINC everything benchmark set.
Table S18. Profile over the number of property occurrences of 43 SMARTS substructures with hydrogen atoms and no recursion in ZINC everything benchmark set.
Table S19. Profile over the number of property occurrences of 39 SMARTS substructures with hydrogen atoms and recursion in ZINC everything benchmark set.
Table S20. Profile over the number of property occurrences of 16 PAINS patterns.
Table S21. Profile for all 2516375 from ZINC lead-like.
Table S22. Profile for all 14059666 form ZINC everything.
Table S23. Profile 61500 molecules selected from ZINC lead-like for the substructure search set.
Table S24. Profile 76800 molecules selected from ZINC everything for substructure search set.
Table S25. Profile first 100000 molecules selected from ZINC lead-like.
Table S26. Ullmann search times in seconds for PAINS substructures as SMARTS in optimized formulation against the complete ZINC lead-like. Scaling factors (SFs) represent the speed up in comparison to the sequential time.
Table S27. VF2 search times in seconds for PAINS substructures as SMARTS in optimized formulation against the complete ZINC lead-like. Scaling factors (SFs) represent the speed up in comparison to the sequential time.
Table S28. Ullmann match times in seconds of the PAINS Substructure Set against the complete ZINC lead-like database. All 16 PAINS are given in the original, an optimized, and an anti-optimized substructure formulation in the SI.
Table S29. VF2 match times in seconds of the PAINS Substructure Set against the complete ZINC lead-like database. All 16 PAINS are given in the original, an optimized, and an anti-optimized substructure formulation in Table 30.
Table S30. SMARTS expressions used in optimization experiment given as taken from literature (original), optimized by the given rule set (optimized) and anti-optimized applying the rule set in reverse (anti-optimized).

Additional file 3: Supplementary Information (Additional file 3).
Figure S1. Depiction of SMARTS pattern with no explicit hydrogens and no recursion (top-left), explicit hydrogens and no recursion (top-right), no explicit hydrogens and recursion (bottom-left), and with explicit hydrogens and recursive atom environments (bottom-right). Depictions are created by SMARTSViewer [42].
Figure S2. Visual dipiction of PAINS patterns 1-4 created with SMARTSViewer [42].
Figure S3. Visual dipiction of PAINS patterns 5-8 created with SMARTSViewer [42].
Figure S4. Visual dipiction of PAINS patterns 9-12 created with SMARTSViewer [42].
Figure S5. Visual dipiction of PAINS patterns 13-16 created with SMARTSViewer [42].

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
H-CE implemented the presented software components, collected the data and performed the comparison studies. MR supervised the project. Both authors read and approved the final manuscript.

Acknowledgements
Many thanks to Angela M. Henzler for revising the manuscript, Karen Schomburg for the help on collecting the SMARTS expressions and Sascha Urbaczek, J. Robert Fischer, Adrian Kolodzik, Tobias Lippert, and Matthias Hilbig for their work on the molecule software components.

Received: 17 February 2012 Accepted: 27 April 2012
Published: 31 July 2012

References

1. Irwin J, Shoichet B: **ZINC–a free database of commercially available compounds for virtual screening.** *J Chem Inf Model* 2005, **45:**177–182.
2. Bolton EE, Wang Y, Thiessen PA, Bryant SH: **Chapter 12 PubChem: Integrated Platform of Small Molecules and Biological Activities.** In *Annual Reports in Computational Chemistry Volume 4, Volume 4 of, Annual Reports in Computational Chemistry*. Edited by Wheeler RA, Spellmeyer DC: Elsevier; 2008:217–241. [http://www.sciencedirect.com/science/article/pii/S1574140008000121]
3. Sussenguth EH: **A graph-theoretic algorithm for matching chemical Structures.** *J Chem Documentation* 1965, **5:**36–43. [http://pubs.acs.org/doi/abs/10.1021/c160016a007]
4. Figueras J: **Substructure search by set reduction.** *J Chem Documentation* 1972, **12**(4):237–244. [http://pubs.acs.org/doi/abs/10.1021/c160047a010]
5. Read RC, Corneil DG: **The graph isomorphism disease.** *J Graph Theory* 1977, **1**(4):339–363. [http://dx.doi.org/10.1002/jgt.3190010410]
6. Gati G: **Further annotated bibliography on the isomorphism disease.** *J Graph Theory* 1979, **3**(2):95–109. [http://dx.doi.org/10.1002/jgt.3190030202]
7. Ullmann JR: **An algorithm for subgraph isomorphism.** *J Assoc Comput Mach* 1976, **23:**31–42.
8. Attias R: **DARC substructure search system: a new approach to chemical information.** *J Chem Inf Comput Sci* 1983, **23**(3):102–108. [http://pubs.acs.org/doi/abs/10.1021/ci00039a003]
9. Heyman J, Karasinskia E, Giles P: **CAS information services for medicinal chemists.** *Drug Inf J* 1982, **16**(4):185–190.
10. Willett P, Barnard JM, Downs GM: **Chemical similarity searching.** *J Chem Inf Model* 1998, **38**(6):983–996. [http://dx.doi.org/10.1021/ci9800211]
11. Cordella L, Foggia P, Sansone C, Vento M: **Performance evaluation of the VF graph matching algorithm.** In *Image Analysis and Processing, 1999. Proceedings. International Conference on*; 1999:1172–1177.
12. Cordella LP, Foggia P, Sansone C, Vento M: **A (sub)graph isomorphism algorithm for matching large graphs.** *IEEE Trans Pattern Anal Machine Intelligence* 2004, **26**(10):1367–1372.
13. Yan X, Yu PS, Han J: **Proceedings of the 2005 ACM SIGMOD international conference on, Management of data, SIGMOD '05.** New York, NY, USA: ACM; 2005:766–777. [http://doi.acm.org/10.1145/1066157.1066244]
14. Golovin A, Henrick K: **Chemical substructure search in SQL.** *J Chem Inf Model* 2009, **49:**22–27.
15. Willett P, Wilson T, Reddaway SF: **Atom-by-atom searching using massive parallelism. Implementation of the Ullmann subgraph isomorphism algorithm on the distributed array processor.** *J Chem Inf Comput Sci* 1991, **31**(2):225–233. [http://pubs.acs.org/doi/abs/10.1021/ci00002a008]
16. Messmer BT: **Efficient Graph Matching Algorithms** 1995.
17. Foggia P, Sansone C, Vento M: **A performance comparison of five algorithms for graph isomorphism.** *Proc of the 3rd IAPR TC-15 Workshop on Graph-based Representations in Pattern Recognition* 2001:188–199.
18. Brint AT, Willett P: **Algorithms For the Identification of 3-dimensional Maximal Common Substructures.** *J Chem Inf Comput Sci* 1987, **27**(4):152–158.
19. Downs GM, Lynch MF, Willett P, Manson GA, Wilson GA: **Transputer implementations of chemical substructure searching algorithms.** *Tetrahedron Comput Methodology* 1988, **1**(3):207–217. [http://dx.doi.org/10.1016/0898-5529(88)90026-7]
20. Barnard JM: **Substructure searching methods: old and new.** *J Chem Inf Comput Sci* 1993, **33**(4):532–538. [http://pubs.acs.org/doi/abs/10.1021/ci00014a001]
21. Oprea TI: *Chemoinformatics in drug discovery*. Weinheim: Wiley-VCH; 2005: 76–79. chap. 4.4.2.1.

22. Agrafiotis DK, Lobanov VS, Shemanarev M, Rassokhin DN, Izrailev S, Jaeger EP, Alex S, Farnum M: **Efficient Substructure Searching of Large Chemical Libraries: The ABCD Chemical Cartridge.** *J Chem Inf Model* 2011, **51:**3113-3130. [http://pubs.acs.org/doi/abs/10.1021/ci200413e]
23. Falkenhainer B, Forbus KD, Gentner D: **The structure-mapping engine: algorithm and examples.** *Artif Intelligence* 1989, **41:**1–63.
24. Tarjan RE: *Graph Algorithms in Chemical Computation*: American Chemical Society; 1977: 1–20. chap. 2. [http://pubs.acs.org/doi/abs/10.1021/bk-1977-0046.ch001]
25. *Daylight Theory Manual, Daylight Chemical Information Systems Inc.* 2011.
26. Ash S, Cline MA, Homer RW, Hurst T, Smith GB: **SYBYL line notation (SLN): A versatile language for chemical structure representation.** *J Chem Inf Comput Sci* 1997, **37:**71–79.
27. Koniver DA, Wiswesser WJ, Usdin E: **Wiswesser line notation: simplified techniques for converting chemical structures to WLN.** *Science* 1972, **176**(4042):1437–1439. [http://dx.doi.org/10.1126/science.176.4042.1437]
28. Hann M, Hudson B, Lewell X, Lifely R, Miller L, Ramsden N: **Strategic pooling of compounds for high-throughput screening.** *J Chem Inf Comput Sci* 1999, **39**(5):897–902. [http://pubs.acs.org/doi/abs/10.1021/ci990423o]
29. Walters W, Murcko MA: **Prediction of 'drug-likeness'.** *Adv Drug Delivery Rev* 2002, **54**(3):255–271. [http://www.sciencedirect.com/science/article/pii/S0169409X02000030]. [Computational Methods for the Prediction of ADME and Toxicity]
30. Abolmaali SFB, Wegner JK, Zell A: **The compressed feature matrix - a fast method for feature based substructure search.** *J Mol Model* 2003, **9:**235–241. [http://dx.doi.org/10.1007/s00894-003-0126-0]. [10.1007/s00894-003-0126-0]
31. Olah M, Bologa C, Oprea TI: **An automated PLS search for biologically relevant QSAR descriptors.** *J Comput Aided Mol Des* 2004, **18:**437–449. [http://dx.doi.org/10.1007/s10822-004-4060-8]. [10.1007/s10822-004-4060-8]
32. Maass P, Schulz-Gasch T, Stahl M, Rarey M: **Recore: a fast and versatile method for scaffold hopping based on small molecule crystal structure conformations.** *J Chem Inf Model* 2007, **47**(2):390–399. [http://pubs.acs.org/doi/abs/10.1021/ci060094h]. [PMID: 17305328]
33. Degen J, Wegscheid-Gerlach C, Zaliani A, Rarey M: **On the art of compiling and using 'drug-like' chemical fragment spaces.** *Chem Med Chem* 2008, **3:**1503-1507.
34. Ahmed HEA, Vogt M, Bajorath J: **Design and evaluation of bonded atom pair descriptors.** *J Chem Inf Model* 2010, **50:**487-499.
35. **Daylight SMARTS examples; Daylight Chemical Information Systems, Inc.** Laguna Niguel, CA; http://www.daylight.com/dayhtml_tutorials/languages/smarts/smarts_examples.html. Accessed May 25, 2010.
36. Agrafiotis DK, Gibbs AC, Zhu F, Izrailev S, Martin E: **Conformational sampling of bioactive molecules: a comparative study.** *J Chem Inf Model* 2007, **47**(3):1067–1086. [http://pubs.acs.org/doi/abs/10.1021/ci6005454]. [PMID: 17411028]
37. Enoch SJ, Madden JC, Cronin MTD: **Identification of mechanisms of toxic action for skin sensitisation using a SMARTS pattern based approach.** *SAR QSAR Environ Res* 2008, **19**(5-6):555–578. [http://dx.doi.org/10.1080/10629360802348985]
38. Baell JB, Holloway GA: **New substructure filters for removal of Pan Assay Interference Compounds (PAINS) from screening libraries and for their exclusion in Bioassays.** *J Med Chem* 2010, **53**(7):2719–2740. [http://pubs.acs.org/doi/abs/10.1021/jm901137j]. [PMID: 20131845]
39. Ihlenfeldt WD, Takahashi Y, Abe H, ichi Sasaki S: **Computation and management of chemical properties in CACTVS: An extensible networked approach toward modularity and compatibility.** *J Chem Inf Comput Sci* 1994, **34:**109–116.
40. Xu J: **GMA: a generic match algorithm for structural homomorphism, isomorphism, and maximal common substructure match and its applications.** *J Chem Inf Comput Sci* 1996, **36:**25–34. [http://pubs.acs.org/doi/abs/10.1021/ci950061u]
41. Gasteiger J, Engel, T (Eds): *Chemoinformatics: A Textbook.* 1 edition. Wiley-VCH; 2003. [http://www.worldcat.org/isbn/3527306811]
42. Schomburg K, Ehrlich HC, Stierand K, Rarey M: **From structure diagrams to visual chemical patterns.** *J Chem Inf Model* 2010, **50**(9):1529–1535. [http://dx.doi.org/10.1021/ci100209a]
43. Ozawa K, Yasuda T, Fujita S: **Substructure search with tree-structured data.** *J Chem Inf Comput Sci* 1997, **37**(4):688–695. [http://pubs.acs.org/doi/abs/10.1021/ci960378%2B]
44. Rughooputh SDDV, Rughooputh HCS: **Neural network based chemical structure indexing.** *J Chem Inf Comput Sci* 2001, **41**(3):713–717. [http://pubs.acs.org/doi/abs/10.1021/ci000394d]
45. Miller MA: **Chemical database techniques in drug discovery.** *Nat Rev Drug Discov* 2002, **1**(3):220–227. [http://dx.doi.org/10.1038/nrd745]
46. Jeliazkova N, Kochev N: **AMBIT-SMARTS: efficient searching of chemical structures and fragments.** *Mol Informatics* 2011, **30**(8):707–720. [http://dx.doi.org/10.1002/minf.201100028]

doi:10.1186/1758-2946-4-13
Cite this article as: Ehrlich and Rarey: Systematic benchmark of substructure search in molecular graphs - From Ullmann to VF2. *Journal of Cheminformatics* 2012 **4**:13.

A2 Searching for substructures in fragment spaces

Authors:	Hans-Christian Ehrlich, Andrea Volkamer, Matthias Rarey
Type of publication:	Journal article
Reference:	Journal of Chemical Information and Modeling, 52(12): 3181–3189, 2012, doi:10.1021/ci300283a
Status:	Published
Legal:	Reprinted with permission from Hans-Christian Ehrlich, Andrea Volkamer, and Matthias Rarey. Searching for substructures in fragment spaces. Journal of Chemical Information and Modeling, 52(12):31813189, 2012. Copyright 2012 American Chemical Society.

JOURNAL OF CHEMICAL INFORMATION AND MODELING

Article

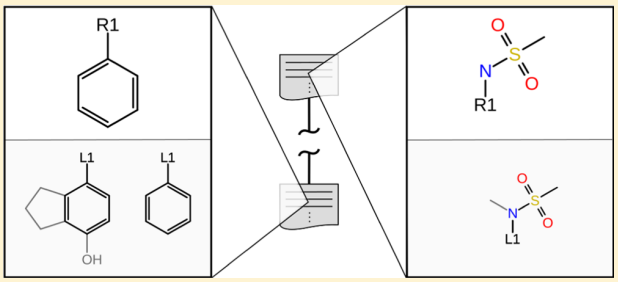
pubs.acs.org/jcim

Searching for Substructures in Fragment Spaces

Hans-Christian Ehrlich, Andrea Volkamer, and Matthias Rarey*

University of Hamburg, Bundestraße 43, 20146 Hamburg, Germany

ABSTRACT: A common task in drug development is the selection of compounds fulfilling specific structural features from a large data pool. While several methods that iteratively search through such data sets exist, their application is limited compared to the infinite character of molecular space. The introduction of the concept of fragment spaces (FSs), which are composed of molecular fragments and their connection rules, made the representation of large combinatorial data sets feasible. At the same time, search algorithms face the problem of structural features spanning over multiple fragments. Due to the combinatorial nature of FSs, an enumeration of all products is impossible. In order to overcome these time and storage issues, we present a method that is able to find substructures in FSs without explicit product enumeration. This is accomplished by splitting substructures into subsubstructures and mapping them onto fragments with respect to fragment connectivity rules. The method has been evaluated on three different drug discovery scenarios considering the exploration of a molecule class, the elaboration of decoration patterns for a molecular core, and the exhaustive query for peptides in FSs. FSs can be searched in seconds, and found products contain novel compounds not present in the PubChem database which may serve as hints for new lead structures.



INTRODUCTION

Finding molecules that fulfill specific structural or physicochemical features is of high practical interest in drug development. Due to the large and still growing number of commercially available and synthetically accessible molecule structures, efficient algorithms for searching large data sets are becoming more and more vital.[1,2] Traditionally, huge compound sets are maintained in large databases. Different computational methods have been developed to efficiently search through these data sets.[3−14] One major application is the retrieval of molecules that include a defined molecular substructure.

To efficiently process large databases, molecules are described as graphs, where nodes denote the atoms and edges the connecting bonds. With such a representation, search algorithms can take advantage of known graph theoretical concepts allowing for an efficient graph comparison. The applied methods range from matrix-based[7] and backtracking algorithms[11,12] for (sub)graph isomorphism, over branch-and-bound,[15,16] maximal clique,[17] and dynamic programming algorithms[18] for maximal common subgraph calculations, to path and radial fragment enumeration[19] for graph similarity search. Nevertheless, since the number of molecules in the chemical universe is almost infinite, databases can reach a critical size where iterative search strategies reach their limits.

Alternative storage principles have been introduced, e.g., Markush structures used in chemical patents. A Markush structure is usually given by a core fragment with open valences and a list of corresponding decoration fragments. A complete molecule is constructed by attaching these fragments to the core until all open attachment points are saturated. A more general concept of such a combinatorial space is a *fragment space* (FS). An FS follows the approach of the retrosynthetical combination analysis procedure (RECAP).[20] RECAP describes distinct rules that model chemical motifs which can easily be formed by combinatorial chemists. An FS is created by applying these rules to separate molecules into fragments. Therefore, an FS consist of molecular fragments with open valences and a set of rules defining their possible combinations to products. For example, the BRICS 4k[21] space comprises 4800 fragments and 64 connection rules. Alternatively, FS can be designed from combinatorial chemistry[22,23] describing reaction schemes in which building blocks are connected prohibiting the formation of cyclic products. Even though the number of fragments in an FS is small, their combinatorial properties allow for the construction of many different products, e.g, enumerating all possible products with up to five fragments in BRICS 4k yields 10^{16} molecules. That is a number which is difficult to handle with a conventional database.

A small number of algorithms to process FSs exist. They solve classical problems in cheminformatics such as the search for similar molecules,[24,25] the novel design of molecules,[26−29] and the creation of FSs focused around target molecules.[30−32]

While methods to search for substructures in molecules exist,[7,11,12] methods browsing through FSs under substructure constraints are rare. Three database systems have been published to search patents for query structures, GENSAL,[33−35] Markush Darc,[36] and MARPAT.[37,38] All systems hold Markush structures retrieved from patent information and store them as reduced graphs. Markush Darc restricts a query to an explicit substructure, whereas MARPAT allows for the occurrence of

Received: June 19, 2012
Published: December 3, 2012

variable and generic groups. Both search methods employ a two step strategy, a screening phase based on limited-environment fragments which is followed by an iterative atom-by-atom search on the remaining structures. The major drawback of both search strategies is their limitation to only handle Markush structures. In more detail, both methods are designed to search the description of core fragments with varying decorations. Unfortunately, the concepts for storing patent information can not, at least without significant modifications, be applied to FSs. An FS consists of rules that may allow for the combination of all fragments and therefore the number of possible products is much higher than when only decorating a core fragment. The main challenge for a substructure search method that processes FSs arise from the possible combination of fragments. A query substructure might not be directly present in any of the fragments but can be constructed by joining two or more fragments into a product. Therefore, the exploration of possible fragment connections leads to a combinatorial large number of products that exceeds the scope of todays computational facilities. Even if a method is able to avoid product enumeration, the search over fragment borders while directly processing connection rules is still a complex task. The only method handling this task is described in a US patent application[39] from 2007. The algorithm uses a modified Ullmann subgraph isomorphism algorithm that assigns parts of the query substructure onto fragments and allows fragment linkers to be assigned to multiple substructure nodes. Multiple node assignment is resolved with respect to the FS connection rules to construct products that contain the full query substructure. Though, the overall algorithmic strategy is similar to our work, neither the modifications to the Ullmann algorithm nor the reconstruction of final products are described in detail.

Here, we present a method for searching substructures in fragment spaces that avoids product enumeration by directly processing fragments and their connection rules. The method finds all products that include a given substructure even if the substructure spans over multiple fragments. The algorithm is designed to minimize the number of explored fragment connections to accomplish reasonable search times. The presented method is evaluated in three tests that mimic different drug development scenarios: the recovery of sulfonamides, a search for new substituents of a kinase inhibitor core, and the retrieval of peptide structures. All three tests are conducted in different FSs, BRICS 4k, BRICS 20k, and the KnowledgeSpace.[40]

PRELIMINARIES

The structural formula is closely related to the mathematical concept of graphs which allows for the direct application of graph theory and algorithms. Therefore, some graph theoretical concepts are introduced in the following.

Graph Theoretical Background. An undirected *graph* $G = (V, E)$ is a set of nodes V and edges E. Each edge $e \in E$ connects two nodes $v_1, v_2 \in V$. Two graph G_1 and G_2 are *isomorphic* if and only if a one-to-one mapping between their nodes V_1 and V_2 exists such that a pair of nodes $v_1, v_2 \in V_1$ is only connected if their images $w_1, w_2 \in V_2$ are connected. An *induced subgraph* of graph $G = (V, E)$ is a graph $G' = (V', E')$ composed of a subset of nodes $V' \subset V$ and edges $E' \subset E$ such that every edge $e = (v_1, v_2) \in E$ connecting two nodes $v_1, v_2 \in V$ is in E' if and only if $v_1, v_2 \in V'$. An *induced subgraph isomorphism* between a query graph G_1 and a target graph G_2 exists if G_1 is isomorphic to an induced subgraph of G_2, i.e., G_2 contains G_1.

A *molecular fragment graph* consist of nodes and edges representing atoms and bonds, respectively. Each edge connects two nodes if a bond connects the corresponding atoms. Nodes are labeled with atomic properties, e.g., atomic symbols, charge, aromaticity, or by an open valence. Two fragments can be combined at open valences to form a larger fragment or a molecule that is a fragment with no open valences. Edges are labeled with bond orders. The number of bonds an atom can form is bound by the atom's valence. Therefore, the node degree of a fragment graph is linearly bound. Note that we will refer to molecular fragment nodes as atoms and to edges as bonds.

A *substructure graph* describes a molecular substructure like a functional group or a molecular core. The graph describes atoms and their connecting bonds by labeled nodes and edges. Again, the node degree is linearly bound by the number of bonds an atom can form. Note that a substructure graph does not allow for the definition of stereochemical centers or alternative mesomeric or tautomeric forms.

METHODS

A method searching for substructures in FSs has to find all combinations of fragments that include the substructure of interest. The presented algorithm divides a query substructure in all possible substructure parts. These *subsubstructures* (SSSs) are searched inside fragments avoiding the combination of fragments to products. From all matches a solution is constructed that describes possible fragment combinations that lead to products incorporating the query substructure. Finally, the algorithm enumerates these products. In the following, each step of the algorithm is explained in detail, as illustrated in Figure 1.

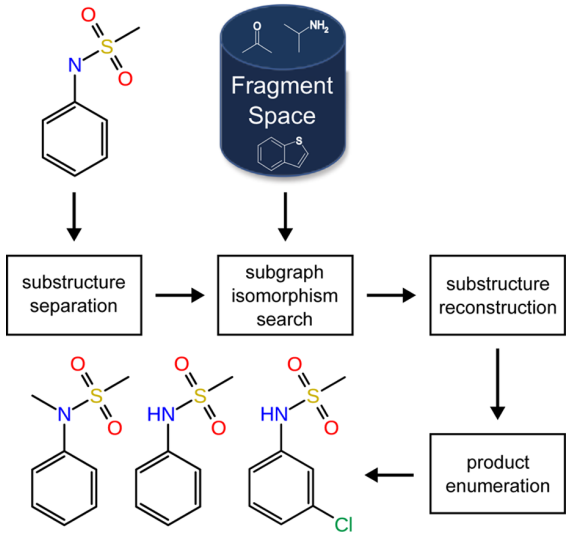


Figure 1. Workflow of matching substructures in fragment spaces (FS). A substructure is separated into subsubstructures. These are matched onto fragments of the FS. On the basis of these matches, recombination trees are constructed which form the basis for product enumeration.

Substructure Separation. A procedure that searches for substructures in FSs faces the problem that substructures might span over multiple fragments. Due to the large number of possible products, the direct examination of fragment connections is undesired. The presented algorithm avoids the combination of fragments during the search phase. It divides the query

dx.doi.org/10.1021/ci300283a | J. Chem. Inf. Model. 2012, 52, 3181−3189

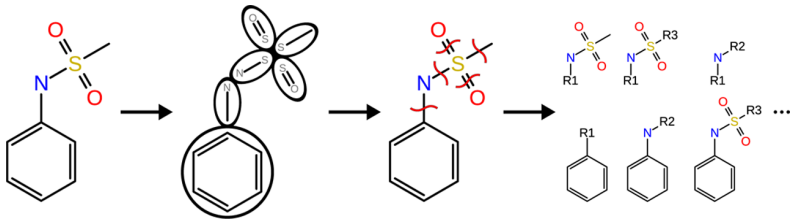


Figure 2. Separation of a query substructure into subsubstructures (SSSs). From the query substructure, cyclic and acyclic components are identified via a biconnected component algorithm. The algorithm assigns cut positions (red) and retrieves SSSs by enumerating all possible cut combinations. The figure only depicts a subset of all possible SSSs.

substructure into subsubstructures. Subsequently, these SSSs are directly mapped onto fragments such that substructure separation points are assigned to fragment linkers. In a first step, the algorithm identifies cut positions that split the query substructure in cyclic and acyclic parts. The following search procedure does not connect fragments into cycles and therefore cyclic substructure parts are not separated. The preservation of cycles is encouraged, since ring formation reactions from combinatorial chemistry usually form the same ring which can be modeled as an independent fragment. The separation algorithm detects cyclic and noncyclic substructure parts using a modified biconnected component (BCC) algorithm.[41] A BCC is either the collection of edges in a cycle or a single acyclic edge. Cut positions are assigned to all acyclic edges except edges to terminal hydrogen nodes. The resulting BCC tree is ordered by a breadth-first-search (BFS) traversal starting from an arbitrary BCC node. On the basis of the BFS order, the method enumerates all possible BCC subtrees using a subtree enumeration algorithm.[42] Since BCC subtree nodes contain substructure edges, an SSS is constructed from the substructure nodes adjacent to edges present in the BCC nodes. Such an SSS represents a part of the original query substructure. Removed substructure parts are indicated by dummy link nodes. Link nodes are labeled such that SSSs can be recombined to the original substructure. Thereby, the method separates the substructure similar to the generation of fragments from molecules. Figure 2 shows a fragmentation example.

Subgraph Isomorphism Search. Since the substructure separation step divides the query substructure into SSSs, the subgraph isomorphism search must be transferred onto the subsubstructure level as well. An FS only allows for a noncyclic connection of fragments. Therefore, a substructure edge that spans over such a connection must also be noncyclic. Since the substructure separation step guarantees that all possible SSSs are generated by splitting the substructure at noncyclic edges, a matching procedure must only find SSSs occurring inside fragments. Later on, these matches are connected to products including the complete query substructure. The modified subgraph isomorphism algorithm[12] matches each SSS against all fragments of the FS (see also Figure 3). According to the described substructure separation procedure, each cut position of an SSSs is marked with a dummy link node. During this search step, SSS nodes are subsequently assigned to fragment atoms until all nodes have a corresponding atom. Dummy nodes are only mapped onto fragment link atoms. Assuming compatible links, matched fragments can be connected at link atoms to form a product in the same way that SSSs can be connected to form the query substructure. The result of the matching phase is a list of matching fragments for each SSS referred to as the *SSS match list*.

Substructure Reconstruction. For the reconstruction of the substructure, the algorithm examines the connectivity of

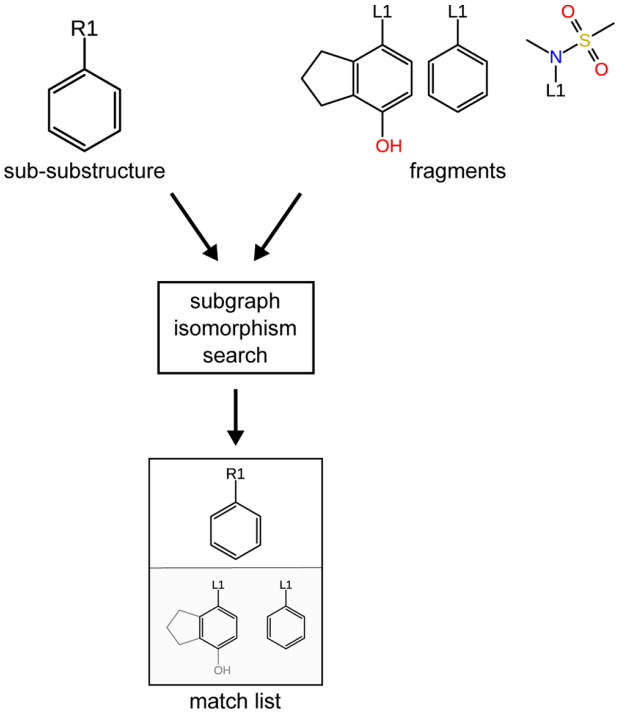


Figure 3. Subgraph isomorphism search example of an SSS that is matched against fragments of the FS. SSS dummy link nodes, labeled with R_1, are matched on fragment link atoms labeled L_1. In this example, an SSS is matched against three fragments. The result is a match list holding the SSS (top) and two matched fragments (bottom).

SSSs and fragments to ensure that the matched fragments can be combined to a product containing the query substructure. A valid combination is described by a *recombination tree*. In such a tree, nodes are represented by SSS match lists. Two lists are connected by an edge if and only if the corresponding SSSs can be connected at dummy nodes and, at the same time, the link atoms matched to the dummy nodes are compatible. In order to achieve a low number of link examinations, the algorithm splits lists that hold fragments with different matched link atoms so that each resulting list holds only fragments with the same link atom types matched to the same dummy nodes. This procedure has the advantage that the link compatibility has to be examined only once, no matter how many fragments are contained in each list. Figure 4 shows an example of a recombination tree which consists of two lists.

Product Enumeration. Each recombination tree represents a substructure separation pattern and holds the corresponding SSSs and fragments in its nodes. The tree's topology describes how SSSs can be connected to form the original query substructure.

dx.doi.org/10.1021/ci300283a | J. Chem. Inf. Model. 2012, 52, 3181−3189

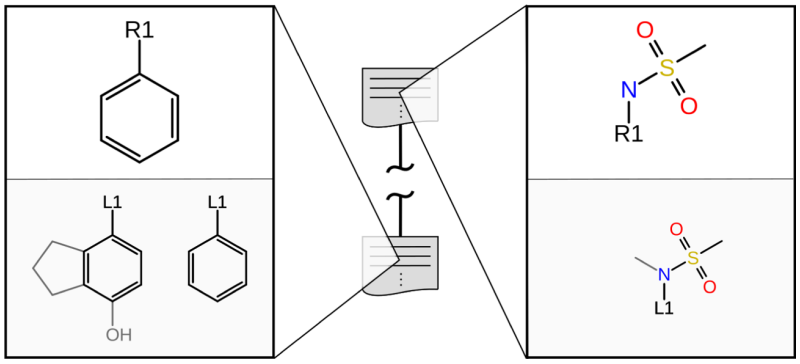


Figure 4. Recombination tree with two SSS match lists. Each list holds an SSS and fragments including that subsubstructure. The original query substructure is constructed by connecting the SSSs at dummy link nodes labeled R_1. A product can be formed by connecting one fragment from each list at the respective link atoms labeled L_1.

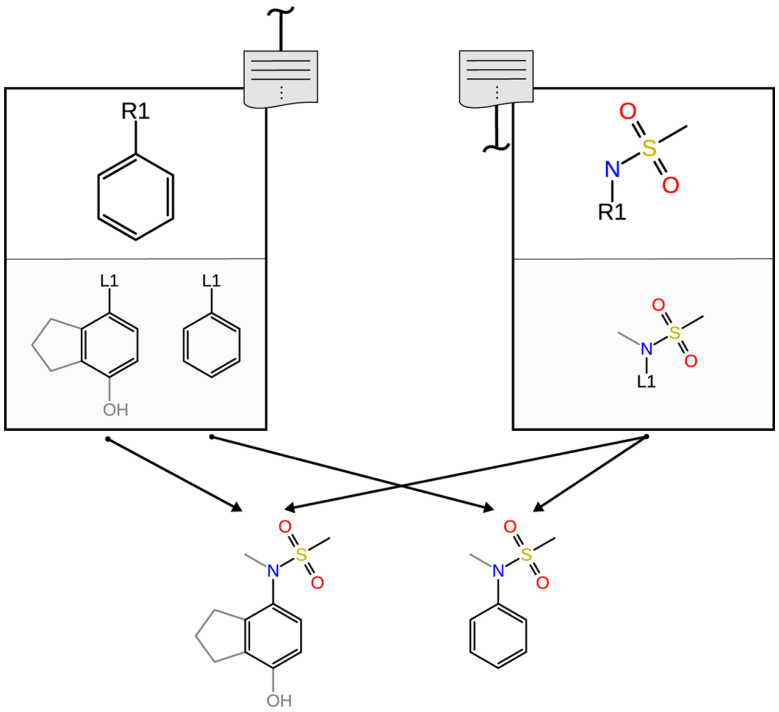


Figure 5. Enumeration of a product from a solution tree. One fragment from each match list of the solution is chosen, and fragments are connected according to their matched link atoms. The result is a molecule or a larger fragment (not shown) that includes the query substructure.

Therefore, fragments from these nodes can be connected accordingly, and the resulting product is guaranteed to include the full query substructure. For enumeration purposes, the algorithm picks one fragment per node and concatenates the fragments according to the connectivity of tree nodes. To avoid multiple enumeration of equal products, the method reduces the fragment lists stored in each node of the recombination tree such that each fragment is contained only once. Fragments are compared with regard to their orientation using a unique number assigned during the FS construction. Additionally, for each generated product the enumeration procedure compares unique SMILES strings.[43] Thereby, equal products arising from different recombination trees are identified and deleted. Repeating this procedure for each recombination tree, the method enumerates the smallest combination of fragments resulting in products including the query substructure. Figure 5 shows a simple enumeration example.

DATA SETS

The efficiency and usefulness of the presented algorithm to search for substructures in FSs is demonstrated in three test scenarios using three publicly available FSs. BRICS 4k and BRICS 20k[21] are generic FSs retrieved from retrosynthetical decomposition of molecules, and KnowledgeSpace[40] is compiled from various synthesis protocols.

The breaking of retrosynthetically interesting chemical substructures (BRICS) approach follows the RECAP concept by describing 16 chemical environments containing different link atoms and 64 rules for connecting them. Figure 6 depicts an example of fragment prototypes and their possible connections. BRICS 4k contains 4800 fragments, and BRICS 20k represents an enrichment of BRICS 4k with an additional 17 200 fragments to include a total of 22 000 building blocks. The BRICS spaces allow the construction of an arbitrary amount of products. A general

dx.doi.org/10.1021/ci300283a | J. Chem. Inf. Model. 2012, 52, 3181−3189

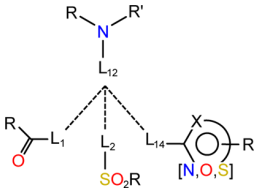


Figure 6. Subset of the BRICS fragment space connection rules. Chemical environments and the corresponding linkers (L) are shown. Omitted parts of environments are indicated with R, and X marks generic atoms. A dotted line between two linkers indicates their compatibility.

measure to describe the size of an FS is the number of possible products that include up to five fragments, which is about 10^{16} for the BRICS 4k space and even more for BRICS 20k.

KnowledgeSpace is based on 82 synthesis protocols obtained from the literature. The protocols cover compounds of specific targets, e.g., GPRCs, proteases, and kinases, as well as purely chemistry-driven substances. KnowledgeSpace comprises 10876 fragments with 488 distinct chemical environments and 7130 connection rules. The chemical space covered reaches about 12×10^9 possible products.

RESULTS

The presented algorithm is tested on three different FSs for its ability to supply alternating molecules of a defined chemical class, different decorations of an arbitrary core, and the extraction of large macromolecules. The measurements include the number of products present in each FS and the search and enumeration times needed on a single Intel(R) Xeon(R) CPU E5630 2.53 GHz core with 64 GB RAM.

EXPLORATION OF A CHEMICAL CLASS

Sulfonamides are the basis for several groups of drugs such as antibacterials, anticonvulsant, and diuretics. Typical sulfonamides are sulfamethoxazole, sulfadoxine, and sulfasalazine. Even though many sulfonamides are known, the search in FSs may reveal novel members with diverse physicochemical properties. Therefore, a substructure search of a sulfonamide defining pattern (Figure 7) against BRICS 4k, BRICS 20k, and KnowledgeSpace, is

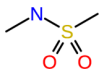


Figure 7. Query substructure defining the class of sulfonamides.

performed. Table 1 gives an overview of the results. The number of single fragments containing the sulfonamide substructure is rather small, e.g., 280 in BRICS 20k. Nevertheless, the combinatorial properties of an FS allow for the construction of almost 10^8 sulfonamides by combining up to three fragments from BRICS 20k.

Due to the chemical environment definition of the respective FSs the maximal product size is limited to three fragments. All three FSs only contain fragments with complete sulfone groups and linking rules that allow the connection of a nitrogen to a sulfone. Furthermore, BRICS contains sulfones with two attached linkers but no linking rule allows the formation of a methyl-sulfone. Therefore, the sulfonamide substructure only spans over three fragments, one of them containing a methyl-sulfone group. KnowledgeSpace allows for the connection of carbon and sulfone but does not contain a sulfone with two attached linkers. Therefore, either the methyl-sulfone or the sulfonamide part must be inside a single fragment. Since all three FSs contain a large number of sulfonamides (Table 1), the potential to find an interesting compound is large. Figure 8

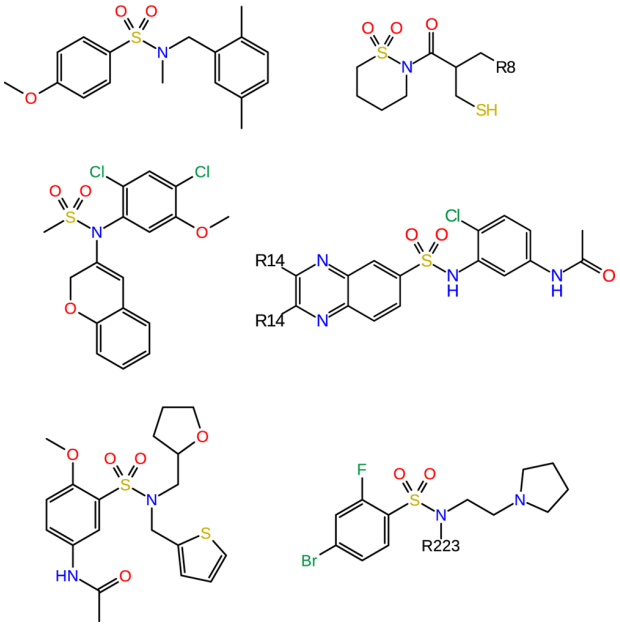


Figure 8. Sulfonamide product examples in BRICS 4k (top), BRICS 20k (middle), and KnowledgeSpace (bottom). Molecules on the left side are present in the PubChem database. Products on the right side are shown with open valences. Open valences allow the attachment of further fragments.

depicts examples from each FS, including commercially available molecules as well as sulfonamides not present in the PubChem database. The second group represents the majority of the retrieved products, showing the potential of the algorithm to find new lead compounds.

Table 1. Search Times, Enumeration Times, and the Number of Products with One, Two, and Three Fragments for a Search of Sulfonamides in BRICS 4k, BRICS 20k, and KnowledgeSpace[a]

			products		
	search time [s]	enum time [m]	1 fragment	2 fragments	3 fragments
BRICS 4k	5.82	2.84	26	3.2×10^4	6.3×10^5
BRICS 20k	31.60	120.08[a]	280[a]	1.3×10^6 [a]	9.3×10^7 [a]
KnowledgeSpace	18.31	49.60	33	3.3×10^4	8.6×10^6

[a]Enumerations stopped at 20 million products due to memory limitations and the number of products is calculated from recombination trees with unique fragments per node (no deduplication by enumeration).

dx.doi.org/10.1021/ci300283a | J. Chem. Inf. Model. 2012, 52, 3181−3189

The substructure search takes between 5 and 32 s to find all fragments that can be combined to a sulfonamide in the respective FS. Enumeration times last from 2.8 min for 63 000 to 2 h for 20 million products. These numbers demonstrate the usability of the search method, especially when considering the large number of products. Enumeration times are 2 orders of magnitude higher in comparison to the search times. An enumeration procedure must account for the possibility that the same product might be generated out of different fragment combinations. Therefore, each enumerated product is checked for uniqueness which is computationally expensive. Nevertheless, the enumeration of 20 million products takes about two hours, which we consider to be acceptable.

The options on how to further process the found products are manifold. They might be subject to further steps in a drug development process, such as similarity queries or molecular property or fingerprint filters. Another option is the construction of a focused FS which itself provides valuable opportunities for lead generation, e.g., allows the use of FS algorithms. In the presented example, a focused FS contains the fragments used in products found during the substructure search and represents a space focused on sulfonamides. For example, a constructed sulfone FS from products found in BRICS 4k contains 2986 fragments, which is 1.6-fold smaller than the original FS and allows an analysis that is more focused on sulfonamides. In general, the large number of products an FS can incorporate makes such an analysis impossible on the set of enumerated products.

MOLECULAR CORE DECORATION

The presented algorithm is well suited for the structure−activity relation exploration of molecular cores. Given the core as substructure, the search procedure generates products containing the core with different decorations. In this experiment, we search for alternatives of Afatinib, shown in Figure 9, developed

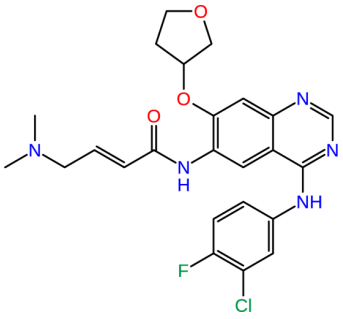


Figure 9. Afatinib structure.

by Boehringer Ingelheim for the treatment of solid tumors. Afatinib is a tyrosine kinase inhibitor. More precisely, it interacts with the epidermal growth factor receptor (EGFR) and the human epidermal growth factor receptor-2 kinases.[44] Figure 10 shows our definition of the basic core of Afatinib. Table 2

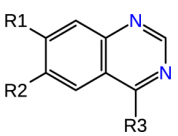


Figure 10. Basic core of Afatinib. The decoration pattern is indicated by the R-groups.

Table 2. Search and Enumeration Times for Unique Decoration of the Afatinib Core in BRICS 4k, BRICS 20k, and KnowledgeSpace

	search time [s]	enum time [s]	products	
			1 fragment	2 fragments
BRICS 4k	4.86	0.72	7	3893
BRICS 20k	39.27	7.93	7	38960
KnowledgeSpace	19.57	0.00	0	0

shows that the search retrieves 3900 and 38 967 different core decorations from BRICS 4k and BRICS 20k, respectively. A closer examination of the results shows that the 38 967 products retrieved from BRICS 20k include all 3900 products from BRICS 4k. This is an expected result, since BRICS 4k resembles a subset of BRICS 20k. KnowledgeSpace does not contain any product with the desired core structure. Figure 11

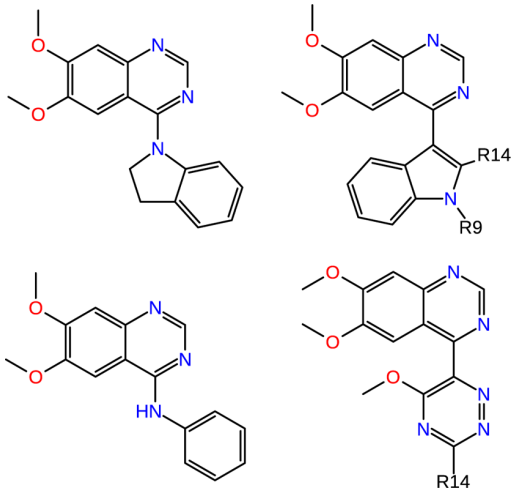


Figure 11. Examples of Afatinib core decorations in BRICS 4k (top) and BRICS 20k (bottom). Molecules on the left side are marked as active against EGFR (top) and EGFR kinases (bottom) in the PubChem database. Products on the right side are shown with open valences. Open valences allow the attachment of further fragments.

shows randomly selected examples from the BRICS fragment spaces. Molecules on the left side are examples documented in PubChem as active against EGFR kinases. A detailed visual inspection of EGFR kinase inhibitors obtained from the PubChem database shows a protonated nitrogen present at R3 of the core definition. A search for a redefined query reveals that 418 out of the 3900 products in BRICS 4k and 3703 from the original 38 967 in BRICS 20k follow this substitution pattern. A query explicitly missing such a protonated hydrogen retrieves the other 3482 and 35 264 products. Therefore, the second query confirms the ability of the search algorithm to retrieve an exact set of products. Search times are in a similar range to the sulfonamide query with 4−40 s and enumeration times of 0.7−8 s are much lower due to the lower number of retrieved products. Again, the found products can be further processed as described in the sulfonamide experiment. For example, a focused FS from BRICS 4k and BRICS 20k would contain 3388 and 16 461 fragments, respectively.

dx.doi.org/10.1021/ci300283a | J. Chem. Inf. Model. 2012, 52, 3181−3189

EXTRACTION OF MACROMOLECULAR STRUCTURES

Oligopeptides are short polymers of amino acids connected by peptide bonds. They are used as inhibitors for kinases, proteases, and HIV-1 assembly.[45] In order to demonstrate the abilities of our method, we search the three FSs for oligopetides with six peptide bonds, shown in Figure 12, requiring an N-terminus and a

Figure 12. Peptide query substructure.

C-terminus for the oligopeptides. 13.3 million and 1.9 billion peptides are found in BRICS 4k and BRICS 20k as shown in Table 3. Figure 13 shows randomly selected examples. Search times of 1.35 min and 9.04 min are achieved on the respective FSs. KnowledgeSpace does not contain any peptide-like products. Since the run time is below 10 min for a search in BRICS 20k, we conclude that even large FSs can be searched with large query substructures in a reasonable time. However, a fingerprint or fragment-based screening step prior to the actual search removing fragments that cannot be part of the final solution would be beneficial for such complex queries. Enumeration times are 3 h for 13 million products and 4.7 h for 20 million products. Nevertheless, run times in the range of hours from a query substructure and a FS to a list of enumerated products render this method very useful. At this point, it should be noted that queries can span over multiple fragments and are not restrained by size limitations.

Interestingly, 45 and 102 fragments form the focused FSs from the search in BRICS 4k and BRICS 20k (see examples in Figure 13), respectively. The low number of fragments show that peptides are formed from few building blocks into billions of products. Therefore, we conclude that both FSs allow the extraction of peptides with arbitrary size by further extending the peptides with more fragments.

CONCLUSIONS

We have presented a novel method that is able to search for substructures in FSs. The method finds all products containing a desired substructure even if the query substructure spans over multiple fragments. The search is not limited in the substructure size or the number of fragments forming a product. The conducted experiments show that the search procedure is fast, below 10 min for a peptide query, especially with respect to the number of matches found and the number of possible products contained in an FS. The computationally most expensive step is the enumeration of products in order to generate a unique set. Regarding the fact that billions of products needed to be compared, we consider a run time of a few hours on a single core acceptable. Since our test products needed to be kept in memory for comparison, memory limitations where encountered at 20 million products. This limitation can be solved by using appropriate database technologies to store and compare enumerated products based on their unique SMILES identifier. Nevertheless, the general applicability and usefulness of the method in a drug development scenario has been demonstrated. The possible

Table 3. Search Times, Enumeration Times, and the Number of Peptides with Six Amide Bonds in BRICS 4k, BRICS 20k, and KnowledgeSpace[a]

			products		
	search time [m]	enum. time [h]	1−5 fragment(s)	6 fragments	7 fragments
BRICS 4k	1.35	3.03	0	5.9×10^5	1.3×10^7
BRICS 20k	9.04	4.72[a]	0	6.4×10^{7a}	1.9×10^{9a}
KnowledgeSpace	5.90	0.00	0	0	0

[a]Enumerations stopped at 20 million products due to memory limitations and the number of products calculated from recombination trees with unique fragments per node (no deduplication by enumeration).

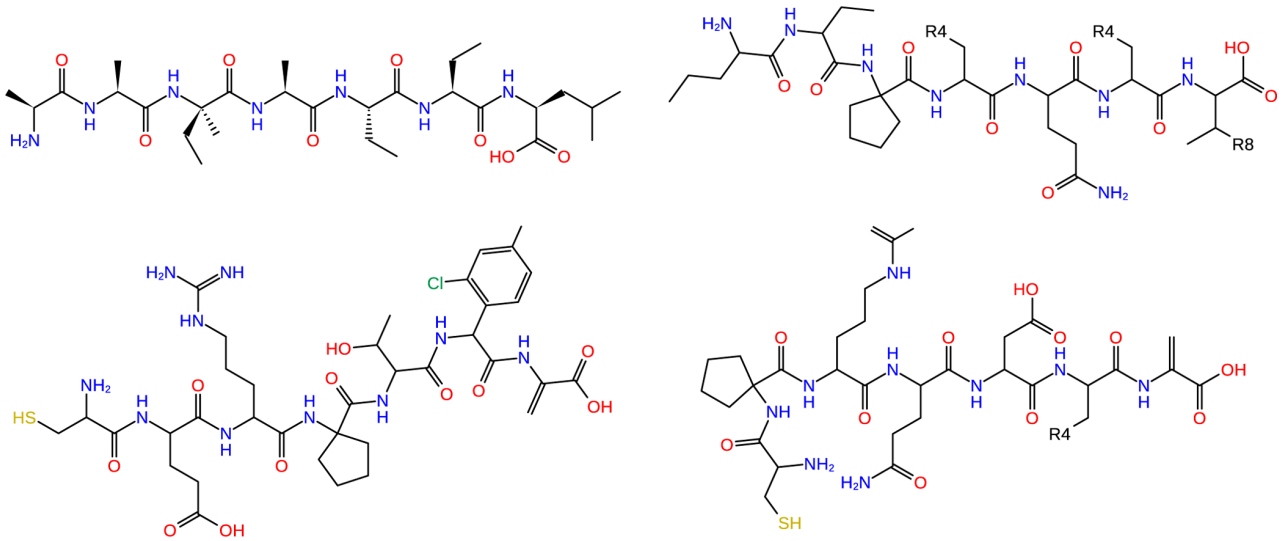


Figure 13. Examples of peptides extracted from BRICS 4k (top) and BRICS 20k (bottom).

dx.doi.org/10.1021/ci300283a | J. Chem. Inf. Model. 2012, 52, 3181−3189

products of an FS might be a valuable source of interesting and novel molecules not contained in publicly available databases. With respect to the number of retrieved matches, products might be visually inspected or subject to further steps in drug development such as analog searches or molecular property filters. A valuable option is the creation of a focused FS from the search results which reduces the number of fragments and focuses the FS for further investigation.

The substructure search method handles explicit substructures quite well. Unfortunately, chiral and generic expressions, such as substructure nodes that match a set of different molecule atoms, alternative tautomeric forms, or atomic properties, e.g., atoms with a defined number of neighbors, are currently not supported. Another limitation of the method is the restriction of cyclic structures occurring only inside fragments. The algorithm will therefore not find structures describing large macromolecular cycles. In most cases, however, the formation of such cyclic structures can be circumvented by a careful FS design. Our future work will extend the search procedure to handle substructure queries with variable atom and bond type definitions as well as logical alternatives, e.g., alternative tautomeric forms, such as present in the Smiles arbitrary target specification (SMARTS).[46]

AUTHOR INFORMATION

Corresponding Author

*E-mail: rarey@zbh.uni-hamburg.de.

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

Thanks to Florian Lauck and others for revising the manuscript.

REFERENCES

(1) Irwin, J.; Shoichet, B. ZINC−a free database of commercially available compounds for virtual screening. *J. Chem. Inf. Model.* **2005**, *45*, 177−182.

(2) Bolton, E. E.; Wang, Y.; Thiessen, P. A.; Bryant, S. H. PubChem: Integrated Platform of Small Molecules and Biological Activities. In *Annual Reports in Computational Chemistry*; Wheeler, R. A., Spellmeyer, D. C., Eds.; Elsevier: New York, 2008; Vol. *4*, Chapter 12, pp 217−241.

(3) Sussenguth, E. H. A Graph-Theoretic Algorithm for Matching Chemical Structures. *J. Graph. Theor.* **1965**, *5*, 36−43.

(4) Figueras, J. Substructure Search by Set Reduction. *J. Graph Theory* **1972**, *12*, 237−244.

(5) Read, R. C.; Corneil, D. G. The graph isomorphism disease. *J. Graph Theory* **1977**, *1*, 339−363.

(6) Gati, G. Further annotated bibliography on the isomorphism disease. *J. Graph Theory* **1979**, *3*, 95−109.

(7) Ullmann, J. R. An algorithm for subgraph isomorphism. *J. Assoc. Comput. Mach.* **1976**, *23*, 31−42.

(8) Attias, R. DARC substructure search system: a new approach to chemical information. *J. Chem. Inf. Comput. Sci.* **1983**, *23*, 102−108.

(9) Heyman, J.; Karasinskia, E.; Giles, P. CAS information services for medicinal chemists. *Drug Inf. J.* **1982**, *16*, 185−190.

(10) Willett, P.; Barnard, J. M.; Downs, G. M. Chemical Similarity Searching. *J. Chem. Inf. Model.* **1998**, *38*, 983−996.

(11) Cordella, L.; Foggia, P.; Sansone, C.; Vento, M. Performance evaluation of the VF graph matching algorithm. In *Image Analysis and Processing, 1999. Proceedings. International Conference on*, Venice, Italy, Sep 27−29; IEEE Computer Society, 1999, pp 1172−1177.

(12) Cordella, L. P.; Foggia, P.; Sansone, C.; Vento, M. A (sub)graph isomorphism algorithm for matching large graphs. *IEEE J. Pattern. Anal.* **2004**, *26*, 1367−1372.

(13) Yan, X.; Yu, P. S.; Han, J. Substructure similarity search in graph databases. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, Baltimore, MD, June 13−17; ACM, New York, 2005; pp 766−777.

(14) Golovin, A.; Henrick, K. Chemical Substructure Search in SQL. *J. Chem. Inf. Model.* **2009**, *49*, 22−27.

(15) Raymond, J. W.; Gardiner, E. J.; Willett, P. RASCAL: Calculation of Graph Similarity using Maximum Common Edge Subgraphs. *Comput. J.* **2002**, *45*, 631−644.

(16) Raymond, J.; Gardiner, E.; Willett, P. Heuristics for similarity searching of chemical graphs using a maximum common edge subgraph algorithm. *J. Chem. Inf. Comput. Sci.* **2002**, *42*, 305−316.

(17) Chao, S.-Y. Maximum Common Substructure Extraction in RNA Secondary Structures Using Clique Detection Approach. *World Acad. Sci., Eng. Technol.* **2008**, *45*, 219−228.

(18) Schietgat, L.; Ramon, J.; Bruynooghe, M.; Blockeel, H. An Efficiently Computable Graph-Based Metric for the Classification of Small Molecules. In *Proceedings of the 11th International Conference on Discovery Science*, Budapest, Hungary, Oct 13−16; Springer-Verlag: Berlin, Heidelberg, Germany, 2008; pp 197−209.

(19) Rogers, D.; Hahn, M. Extended-connectivity fingerprints. *J. Chem. Inf. Model.* **2010**, *50*, 742−754.

(20) Lewell, X. Q.; Judd, D. B.; Watson, S. P.; Hann, M. M. RECAP−retrosynthetic combinatorial analysis procedure: a powerful new technique for identifying privileged molecular fragments with useful applications in combinatorial chemistry. *J. Chem. Inf. Comput. Sci.* **1998**, *38*, 511−522.

(21) Degen, J.; Wegscheid-Gerlach, C.; Zaliani, A.; Rarey, M. On the art of compiling and using 'drug-like' chemical fragment spaces. *ChemMedChem* **2008**, *3*, 1503−1507.

(22) Boehm, M.; Wu, T.-Y.; Claussen, H.; Lemmen, C. Similarity searching and scaffold hopping in synthetically accessible combinatorial chemistry spaces. *J. Med. Chem.* **2008**, *51*, 2468−2480.

(23) Lessel, U.; Wellenzohn, B.; Lilienthal, M.; Claussen, H. Searching Fragment Spaces with feature trees. *J. Chem. Inf. Model.* **2009**, *49*, 270−279.

(24) Rarey, M.; Stahl, M. Similarity searching in large combinatorial chemistry spaces. *J. Comput.-Aided Mol. Des.* **2001**, *15*, 497−520.

(25) Schneider, G.; Neidhart, W.; Giller, T.; Schmid, G. "Scaffold-Hopping" by Topological Pharmacophore Search: A Contribution to Virtual Screening. *Angew. Chem., Int. Ed. Engl.* **1999**, *38*, 2894−2896.

(26) Schneider, G.; Clément-Chomienne, O.; Hilfiger, L.; Schneider, P.; Kirsch, S.; Böhm, H.-J.; Neidhart, W. Virtual Screening for Bioactive Molecules by Evolutionary De Novo Design. *Angew. Chem., Int. Ed.* **2000**, *39*, 4130−4133.

(27) Hartenfeller, M.; Proschak, E.; Schüller, A.; Schneider, G. Concept of combinatorial de novo design of drug-like molecules by particle swarm optimization. *Chem. Biol. Drug. Des.* **2008**, *72*, 16−26.

(28) Lippert, T.; Schulz-Gasch, T.; Roche, O.; Guba, W.; Rarey, M. De novo design by pharmacophore-based searches in fragment spaces. *J. Comput.-Aided Mol. Des.* **2011**, *25*, 931−945.

(29) Hartenfeller, M.; Zettl, H.; Walter, M.; Rupp, M.; Reisen, F.; Proschak, E.; Weggen, S.; Stark, H.; Schneider, G. DOGS: reaction-driven de novo design of bioactive compounds. *PLoS Comput. Biol.* **2012**, *8*, e1002380.

(30) Good, A. C.; Lewis, R. A. New methodology for profiling combinatorial libraries and screening sets: cleaning up the design process with HARPick. *J. Med. Chem.* **1997**, *40*, 3926−3936.

(31) Gillet, V. J.; Willett, P.; Fleming, P. J.; Green, D. V. S. Designing focused libraries using MoSELECT. *J. Mol. Graphics Modell.* **2002**, *20*, 491−498.

(32) Fischer, J.; Lessel, U.; Rarey, M. LoFT: Similarity-Driven Multiobjective Focused Library Design. *J. Chem. Inf. Model.* **2010**, *50*, 1−21.

(33) Barnard, J. M.; Lynch, M. F.; Welford, S. M. Computer storage and retrieval of generic chemical structures in patents. 2. GENSAL, a formal language for the description of generic chemical structures. *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 151−161.

(34) Lynch, M. F.; Holliday, J. D. The Sheffield Generic Structures Projecta Retrospective Review. *J. Chem. Inf. Comput. Sci.* **1996**, *36*, 930−936.
(35) Downs, G. M.; Barnard, J. M. Chemical patents and structural information - the Sheffield research in context. *J. Doc.* **1998**, *54*, 106−120.
(36) Benichou, P.; Klimczak, C.; Borne, P. Handling Genericity in Chemical Structures Using the Markush Darc Software. *J. Chem. Inf. Comput. Sci.* **1997**, *37*, 43−53.
(37) Fisanick, W. The Chemical Abstract's Service generic chemical (Markush) structure storage and retrieval capability. 1. Basic concepts. *J. Chem. Inf. Comput. Sci.* **1990**, *30*, 145−154.
(38) Ebe, T.; Sanderson, K. A.; Wilson, P. S. The Chemical Abstracts Service generic chemical (Markush) structure storage and retrieval capability. 2. The MARPAT file. *J. Chem. Inf. Comput. Sci.* **1991**, *31*, 31−36.
(39) Domine, D.; Cedric, M. *Method for fast substructure searching in non-enumerated chemical libraries*. US Patent Application US 2007/0260583 A1, 2007.
(40) Detering, C.; Claussen, H.; Gastreich, M.; Lemmen, C. KnowledgeSpace - a publicly available virtual chemistry space. *J. Cheminf.* **2010**, *2*, O9.
(41) Hopcroft, J.; Tarjan, R. Algorithm 447: efficient algorithms for graph manipulation. *Commun. ACM* **1973**, *16*, 372−378.
(42) Rarey, M.; Dixon, J. S. Feature trees: a new molecular similarity measure based on tree matching. *J. Comput.-Aided Mol. Des.* **1998**, *12*, 471−490.
(43) Weininger, D.; Weininger, A.; Weininger, J. L. SMILES 2. Algorithm for Generation of Unique SMILES Notation. *J. Chem. Inf. Comput. Sci.* **1989**, *29*, 97−101.
(44) Minkovsky, N.; Berezov, A. BIBW-2992, a dual receptor tyrosine kinase inhibitor for the treatment of solid tumors. *Curr. Opin. Investig. Drugs* **2008**, *9*, 1336−1346.
(45) Owens, R. J.; Tanner, C. C.; Mulligan, M. J.; Srinivas, R. V.; Compans, R. W. Oligopeptide inhibitors of HIV-induced syncytium formation. *AIDS Res. Hum. Retroviruses* **1990**, *6*, 1289−1296.
(46) *Daylight Theory Manual*, version 4.9; Daylight Chemical Information Systems Inc.: Aliso Viejo, CA, 2008.

A3 Searching for recursively defined substructures in non-enumerated fragment spaces

Authors:	Hans-Christian Ehrlich, Angela M. Henzler, Matthias Rarey
Type of publication:	Journal article
Reference:	Journal of Chemical Information and Modeling, 53(7), 1676–1688, 2013, doi:10.1021/ci400107k
Status:	Published

JOURNAL OF CHEMICAL INFORMATION AND MODELING

Article

pubs.acs.org/jcim

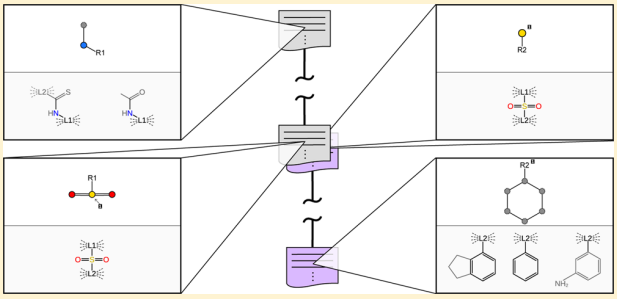
Searching for Recursively Defined Generic Chemical Patterns in Nonenumerated Fragment Spaces

Hans-Christian Ehrlich, Angela M. Henzler, and Matthias Rarey*

University of Hamburg, Bundestraße 43, 20146 Hamburg, Germany

Supporting Information

ABSTRACT: Retrieving molecules with specific structural features is a fundamental requirement of today's molecular database technologies. Estimates claim the chemical space relevant for drug discovery to be around 10^{60} molecules. This figure is many orders of magnitude larger than the amount of molecules conventional databases retain today and will store in the future. An elegant description of such a large chemical space is provided by the concept of fragment spaces. A fragment space comprises fragments that are molecules with open valences and describes rules how to connect these fragments to products. Due to the combinatorial nature of fragment spaces, a complete enumeration of its products is intractable. We present an algorithm to search fragment spaces for generic chemical patterns as present in the SMARTS chemical pattern language. Our method allows specification of the chemical surrounding of an atom in a query and, therefore, enables a chemically intuitive search. During the search, the costly enumeration of products is avoided. The result is a fragment space that exactly describes all possible molecules that contain the user-defined pattern. We evaluated the algorithm in three different drug development use-cases and performed a large scale statistical analysis with 738 SMARTS patterns on three public available fragment spaces. Our results show the ability of the algorithm to explore the chemical space around known active molecules, to analyze fragment spaces for the presence of likely toxic molecules, and to identify complex macromolecular structures under additional structural constraints. By searching the fragment space in its nonenumerated form, spaces covering up to 10^{19} molecules can be examined in times ranging between 47 s and 19 min depending on the complexity of the query pattern.



INTRODUCTION

Various *in silico* applications in drug discovery are confronted with the exploration of molecular databases. Often, the identification of specific query molecules or the filtering of databases according to predefined structural properties is of interest. Hence, most chemical software tools support database searches employing molecule, substructure, or chemical pattern matching algorithms. With increasing size of molecular databases, conventional molecule and especially chemical pattern search[1−12] becomes demanding concerning storage and search time requirements. With some exceptions, e.g., the Chemical Universe Database GDB-13, which comprises around 970 million entries,[13] molecular databases generally store only a few million compounds.[14,15] However, the size of the chemical space is estimated to be much larger. Reported numbers range between 10^{18} and 10^{200} with a consensus around 10^{60} possible structures,[16−18] amounts that by far exceed the critical limit of storage capacities provided by conventional databases. In order to allow a sampling of the chemical space, efficient storage and search strategies are required. Two concepts for this exist: fragment spaces[19−22] (FS's) and Markush structures.[23−26] Both allow a very compact description by an efficient graph-based representation of a large chemical space. Moreover, they enable the application of efficient graph-matching algorithms for molecular search.[27−31]

Fragment spaces are described by molecular fragments and connection rules. In this context, a molecular *fragment* is a graph composed of nodes and edges that represent atoms and bonds. Nodes are labeled with atomic properties, e.g., chemical symbol, charge, or open valence, whereas edges are labeled with bond orders. The number of edges attached to one node is linearly limited by the number of bonds an atom can form. Fragments possess *link atoms* that indicate open valences. Larger fragments, which we refer to as products, or molecules can be built by linking fragments according to their connection rules. Molecules are products without link atoms. The connection rules of an FS define how link atoms of different types can be combined and, thus, which products are encoded by the FS. Fragment spaces are obtained by retrosynthetic cleavage of complete molecules and forming a consensus on the resulting fragments. They intend to reflect the knowledge about reaction schemes of combinatorial chemistry. The combinatorial nature of FS's allows description of a large number of molecules with only a few fragments and rules; for example, the

Received: February 14, 2013
Published: June 11, 2013

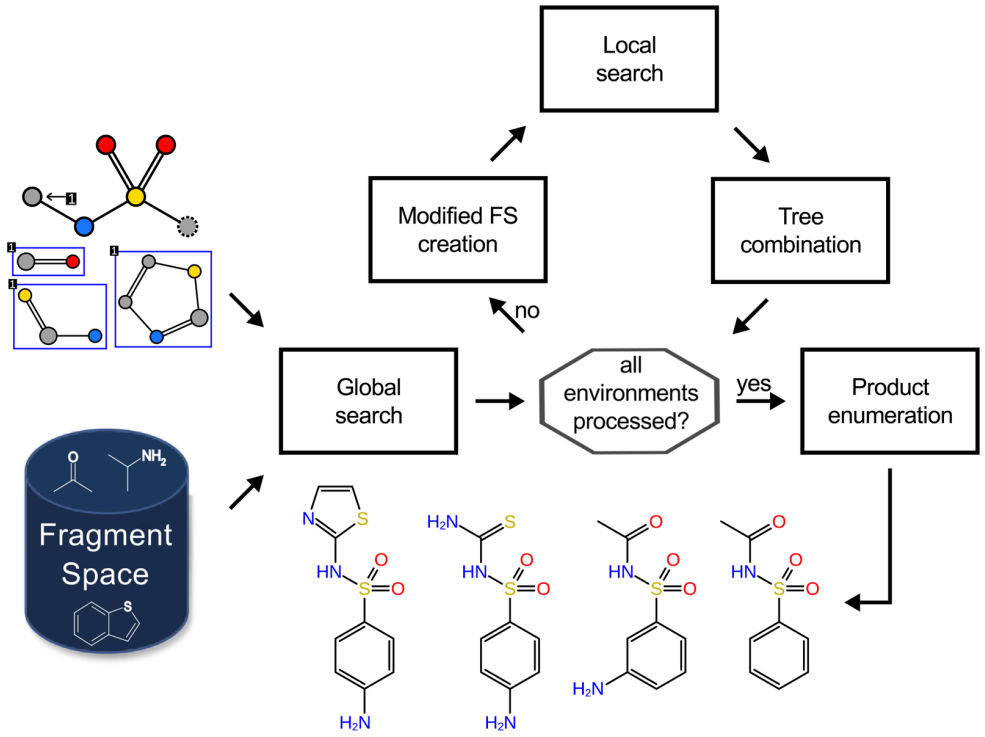


Figure 1. Pattern search for sulfonamides under additional structural constraints. The method utilizes a global search that scans the FS for the pattern neglecting all atomic environments. The algorithm subsequently adds all environments. In an iterative procedure, a modified FS is created for each recombination tree obtained from the global search. These spaces are searched for an environment pattern in a local search. The global tree is modified using the local search results, which generates multiple trees. The procedure is repeated for all trees until all recursive environments are fulfilled. From the final result, molecular products are enumerated to obtain complete molecules. (Substructure depiction created with SmartsViewer.[47])

BRICS 4k space[21] containing 4800 fragments and 64 connection rules encodes over 10^{16} molecules.

The concept of *Markush structures* is quite similar to that of FS's. Markush structures are designed to represent a series of homologous molecules. They usually consist of a single core fragment containing R-groups that are open attachment points. For each group, a list specifies alternative fragments or generic structures, e.g., any alkene. In general, it is undesired to explicitly enumerate the space encoded by Markush structures. Thus, database systems storing Markush structures, e.g. GENSAL,[29] Markush Darc,[30] and MAPRAT,[28] process and search entries avoiding an explicit enumeration.

Even though both concepts are quite similar, there is still a need for exact molecule, substructure, and chemical pattern search methods on FS's. Methods that process and search databases of Markush structures are not transferable to FS's. The main reason is that algorithms on FS's require completely resolved atoms of the fragments. This is contradictory to the concept of generic substituents in Markush structures, as they do not always have an explicit structural counterpart in the graph description. Current algorithms operating on FS's support mainly similarity searching,[31,32] the design of novel molecules,[33−36] and the creation of focused compound libraries[37−39] for virtual screening campaigns. Procedures for substructure search in FS's are rare. Domine and Cedric applied for a patent for substructure searches in nonenumerated chemical spaces that use a modified Ullmann algorithm to assign parts of the substructure to fragments which are subsequently reassembled into products.[40] However, the patent description misses details of the algorithmic concepts. We recently introduced a similar method for substructure search in FS's.[41] It recognizes the fact that a substructure can span over multiple fragments and avoids an explicit enumeration of products during the search.

The SMiles ARbitrary Target Specification (SMARTS),[42] as well as other *chemical pattern languages* such as the Sybyl Line Notation[43] or the Molecular Query Language (MQL),[44] defines properties of atoms and bonds and the topology of chemical patterns. Provided that the query in the form of a textual line notation is transferred to a graph, chemical pattern searches can benefit from graph matching algorithms.[5,9,10,45] Graphs describing such chemical patterns are further referred to as *pattern graphs*. As opposed to graphs representing molecules or substructures, the nodes and edges can be labeled with a generic description of atoms and bonds. The generic description accounts for a broader spectrum of atom and bond properties which can be combined in logical expressions. A useful and practical feature of the SMARTS language is its ability to specify recursive *atomic environments*. They describe the molecular surrounding of an atom, e.g., a nitrogen that is part of a sulfonamide group, and they are well suited to describe local alternatives such as mesomeric and tautomeric forms. Chemical atomic environments can be modeled as pattern graphs assigned to nodes of a higher level pattern graph. Due to the recursive nature of atomic environments, chemical patterns can be nested in patterns. As a result, an exact pattern search has to recursively traverse nested patterns in order to identify matching molecules.

dx.doi.org/10.1021/ci400107k | J. Chem. Inf. Model. 2013, 53, 1676−1688

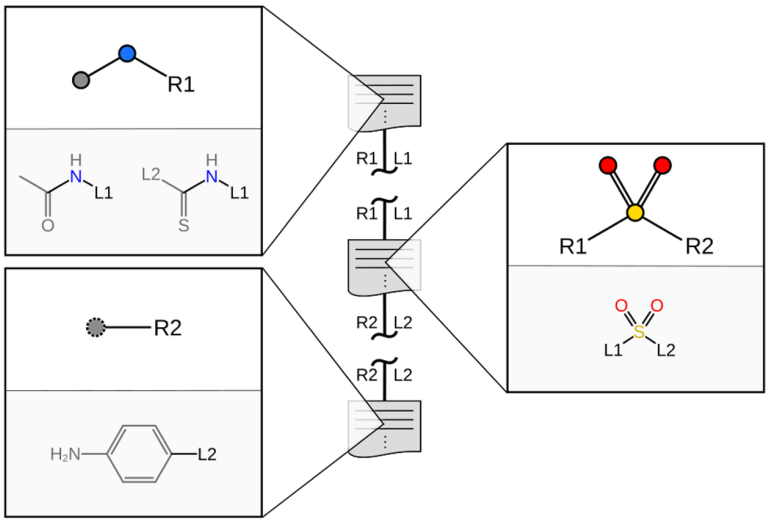


Figure 2. Recombination tree gained from a sulfonamide query search. The tree contains three nodes, each holding an SP, corresponding fragments, and the assignment of the SP to the fragments. The SP's are compatible at dummy link node *R*1, respectively *R*2, to form the sulfonamide query pattern. Fragments can be connected via linker *L*1, respectively *L*2, to form a sulfonamide product.

In this paper, we introduce the extension of our substructure search algorithm for nonenumerated FS's[41] with regard to SMARTS queries with recursive atomic environments. In order to promote a profound understanding, we initially overview the workflow of our new method and briefly describe the terminology and the basic concept of our previously introduced search procedure. We continue with a step-by-step explanation of the adaptations and extensions necessary to account for atomic environments. The method is evaluated by three use-case studies that simulate different molecular modeling scenarios. Each of them is conducted on the BRICS 4k, the BRICS 20k,[21] and the KnowledgeSpace[46] FS's. The results demonstrate that the method creates subspaces in which all molecules contain a specific chemical pattern. Concluding studies on two publicly available data sets finally reflect the run time behavior of our new method. At the current stage, a cyclic connection of fragments and stereoisomeric chemical patterns is not addressed by the algorithm.

METHOD

The search strategy for patterns with recursive atomic environments follows the workflow depicted in Figure 1. In a first step, named *global search*, the algorithm processes the *global pattern* which is the query pattern neglecting the atomic environment information. The obtained global result is subsequently modified by *atomic environment searches*. An atomic environment is a pattern that defines the chemical surrounding of an atom expressed as recursive SMARTS. In the following, we will refer to such an atom as the *reference atom* of the environment. When searching in FS's, such an atomic environment either can be found in fragments itself or might be present in a combination of fragments. In the latter case, the atomic environment can only be indirectly identified by attaching additional fragments. In order to determine whether an atomic environment exists and which case occurs without an enumeration of products, the atomic environment search follows a recurring three step procedure: a modified FS is created from the global result, the new space is scanned for the atomic environment pattern, and the obtained results are combined. The search is repeated until all atomic environments are fulfilled or no acceptable combination of fragments can be found. Both searches utilize the *generic search strategy* to scan FS's for patterns without atomic environment information. At the end, the method enumerates the actual products.

Generic Search Strategy. In FS's a pattern can span over multiple fragments. To avoid a time-consuming exploration of fragment connections during the pattern search, the generic search follows the previously described strategy[41] by separating the query pattern into subpatterns (SP's), similarly to the separation of molecules into fragments. An SP is a connected part of the original pattern in which cyclic parts are fully contained and missing pieces of the pattern are indicated by dummy link nodes. The actual subgraph isomorphism step is a search that matches SP's against fragments and assigns dummy link nodes to fragment link atoms, avoiding the costly exploration of fragment link connections. The method records a list of matching fragments for each SP. Since the fragments of a list might have different link types assigned to the same dummy node, the lists are split to obtain smaller lists with unique underlying fragment linkers. This procedure has the advantage that the link compatibility has only to be checked once, no matter how many fragments are stored in each list. In preparation for the reconstruction process of molecules containing the complete query pattern, these lists are used as nodes to build a *recombination tree*. Each node stores fragments, an SP, and its possible assignments to fragments. In a recombination tree, two nodes are connected if and only if their SP's can be connected according to the connectivity of the query pattern and if the fragment link atoms are compatible according to the connection rules of the FS. The nodes are connected by two half edges. Each half is labeled with the dummy node of the SP and the corresponding link atoms. Therefore, a recombination tree describes a separation of the pattern and the associated combination of fragments. Since a pattern can be separated in various ways which lead to different sets of SP's, the result of the search is a collection of recombination trees describing a subspace of the searched FS. Figure 2 shows an example of a recombination tree.

Global Search. The global search scans an FS for the presence of the global pattern. It utilizes the generic search

strategy to gain a set of global recombination trees that describe combinations of fragments that include the global pattern but omits the atomic environment information. The environments of reference atoms are resolved by atomic environment searches.

Atomic Environment Search. Atomic environments of reference atoms occurring in the resulting trees are resolved by a three-step iterative procedure: for each combination of an atomic environment and a global recombination tree, a modified FS is constructed. The new FS enforces the combinations of fragments described by the tree. After the creation, the space is scanned with the local search strategy for the respective environment pattern. The result is a set of local recombination trees. As described below, each local tree is combined with the global tree to describe matches that include the global pattern and obey the environment information. The method repeats this process of FS creation, search, and result combination until all atomic environments are processed. The final result is a collection of recombination trees describing a portion of the FS that includes products incorporating the query pattern with regard to the environment information.

The SMARTS pattern language allows the use of atomic environments in combination with logical expressions such as logical NOT, AND, OR, and WEAK-AND. A logical WEAK-AND is a SMARTS specific term that serves as a replacement for a logical grouping of AND and OR terms; for example, the term "*A* WEAK-AND *B* OR *C*" represents "*A* AND (*B* OR *C*)". Such a definition is internally resolved into a disjunctive normal form (DNF) that only contains logical NOT, AND, and OR, e.g., "*A* AND *B* OR *A* AND *C*". In addition, each ANDed term is sorted such that non-negated environments precede negated environments. In the following, we describe the iteratively performed modification of a single global recombination tree depending on the logical markup given by the DNF. The other trees of the global result are accordingly modified. Figure 4 gives a schematic example.

The highest logical level of a DNF is a logical OR that combines logical terms in which negated and non-negated atomic environments are exclusively ANDed. If logical terms are ORed, they enforce the presence of an environment described by at least one of the terms. For each ORed term, the global recombination tree is duplicated and modified according to the logical AND description of the term. Therefore, the duplicated tree is updated to obey the atomic environments of the term. The final result is a collection of recombination trees describing the smallest combinations of fragments that include the query pattern and fulfill the logical specification of all atomic environments.

One level down in a DNF is a logical AND connecting, potentially with logical NOT negated, atomic environments. If environments are logically ANDed, the global recombination tree must be modified to simultaneously obey multiple atomic environments. The procedure processes environments in the essential order of non-negated followed by negated environments. For every non-negated environment, the inclusion of a single environment results in a modified global recombination tree in which fragments that contradict the AND description are removed from the nodes or additional fragments are attached to ensure that the environment is present. This inclusion modification is repeated on the resulting tree for each non-negated environment. If environments are negated, they have to be explicitly excluded from the recombination trees. The exclusion of a single environment from a tree results in multiple recombination trees. For each tree resulting from an exclusion, the next environment is excluded until all negated environments are processed. The result is a set of recombination trees that describe products including the global pattern and all logically ANDed environments.

In the following, we explain one iteration of the atomic environment search for the modification of a recombination tree to obey a single negated or non-negated environment specification. Recursively defined environments that define the surrounding of a reference atom with another atomic environment enforce an additional search. In such a case, the current local pattern is treated as the global pattern and the additional environment pattern is subjected to the atomic environment search.

Modifying the Fragment Space. A pattern that spans over multiple linkers might define an atomic environment that also spans over a number of fragments. A search method must therefore detect the common set of fragments and link atoms between trees obtained from the global and local search to decide whether the atomic environment exists or if compatible fragments need to be added to satisfy the atomic environment specification. This problem is addressed by creating a modified FS from a recombination tree. Such a space includes all fragments and link rules from the original space. In addition, the space contains all fragments of the tree in a modified fashion: the matched link atoms of its matching fragments are renamed, and connection rules are added that only allow the combination of these fragments according to the recombination tree. Unmatched link atoms are also renamed, and rules that prohibit a linkage to other matched fragments are formulated. A connection to unmatched fragments according to the original link connections is still possible to allow the further attachment of fragments. The renaming scheme allows a clear differentiation between matched and unmatched linkers. Figure 3 illustrates the process of constructing a modified FS.

Local Search. After constructing the modified FS, the method searches for the associated atomic environment pattern in the modified space. For the atomic environment under consideration, first all reference atoms have to be identified using the recombination tree. The search starts at such an atom in the corresponding fragment and proceeds using the generic search to detect the atomic environment pattern in the modified FS. The result is a set of local recombination trees which can be empty, indicating that the environment was not found. Otherwise, each local tree of the set contains modified fragments from the global recombination tree or unmodified fragments. If fragments from the global tree are identified, a connection to other fragments is only possible via renamed linkers. Since the renaming allows a clear differentiation between matched and unmatched linkers, the common set of fragments and linkers in the global and local recombination trees can be detected, which allows the following combination of results.

Combining Recombination Trees. From the global and the local search the obtained matches are represented by a global recombination tree and a set of local recombination trees. The aim of the next step is to combine these recombination trees such that they describe the matching of the global pattern including the atomic environment represented by the local trees. If the set of local recombination trees is empty and the corresponding atomic environment is not negated, the global tree is discarded because the atomic environment was not found. If the environment is negated, the global tree is not

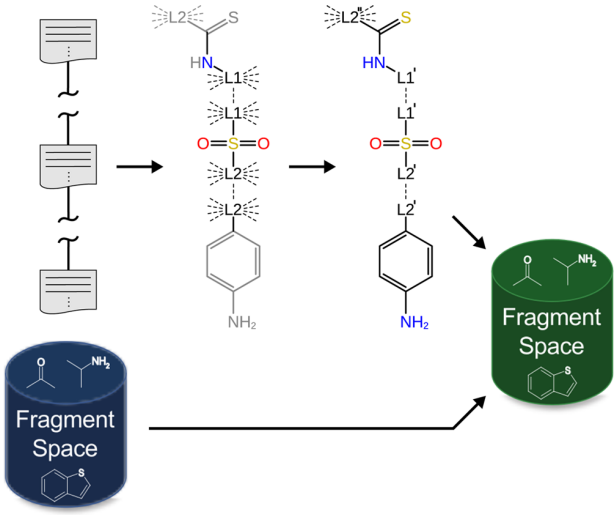


Figure 3. Modification of an FS after a sulfonamide search. The original FS is enriched with three fragments obtained from the recombination tree. These fragments are modified such that the matched link atoms L_1 and L_2 are renamed to L_1' and L_2' and a connection rule is added that only allows a connection of the two shown fragments to form a sulfonamide. The unmatched link atom L_2 is renamed to L_2'' such that a linkage to matched fragments is prohibited and connections to other fragments are still allowed; for example, if $L2$ is compatible to $L1$, the new FS allows a connection of $L2''$ to $L1$ but not to $L1'$. Note that all products previously included in the FS are still contained in the modified FS, since no fragments or rules are removed.

modified. Otherwise, each local recombination tree is subsequently superimposed with the global recombination tree as follows: both trees share at least one common fragment, the fragment containing the reference atom. Initially, the associated pair of tree nodes are superimposed, and subsequently, their incident edges are searched for equally renamed linkers. This procedure ensures that both the global and the local match use identical linkers to attach the same fragments. The superimposition proceeds for adjacent nodes containing fragments that can be attached by equally renamed linkers. If linkers have no appropriate counterpart, the node is not superimposed. Figure 4 shows an example for the superimpositions of recombination trees.

For each superimposition of the global and a single local recombination tree, the procedure modifies the global tree depending on the logical markup of the atomic environment pattern. If an environment is non-negated, the molecular surrounding of the reference atom must be included in the global recombination tree. An *inclusion* demands that the global recombination tree is modified to describe combinations of fragments that include the global pattern and the atomic environment pattern at the same time. To generate such a tree, the procedure intersects the set of matched fragments associated with superimposed nodes and stores the resulting set in the corresponding global tree node. If the resulting set is empty, the global tree is discarded. Local tree nodes not superimposed are attached to the global recombination tree in order to add fragments that fulfill the atomic environment specification.

If an atomic environment is negated, it must not exist around the reference atom. In order to *exclude* it, multiple recombination trees are generated. For each local tree node, the global tree is duplicated. The procedure modifies this tree copy according to the currently chosen local tree node: if it was previously superpositioned, the set exclusion of the local fragments from the global fragments is stored in the corresponding node of the tree copy. If the exclusion generates an empty set, the tree copy is discarded. Otherwise, the node stores only fragments that include the global SP but do not contain the atomic environment SP. Consequently, the full environment is not formed when the fragments of the tree copy are connected. To exclude the environment of a node that is not superimposed, a new node is attached to the tree copy that stores an inversion of the matched fragments present in the current local tree node. A set of fragments is inverted by excluding the set from all fragments of the FS that are compatible with the current linker. Again, if the inversion results in an empty set, the tree copy is not further considered. If such a node is not directly attached to a superimposed node, the path to this node is attached to the copy of the global tree and the set of fragments of the node is inverted.

Creating a new tree for every node related to the excluded atomic environment generates all trees that describe matches that do not include the negated pattern. Even if additional fragments are attached to these products, the environment is guaranteed to not emerge around the reference atom.

Example: Atomic Environment Search. Figure 4 shows four iterations of the atomic environment search. In this example, the global search resulted in a global recombination tree (white). This tree is extended to follow the logical expression "*X* AND *Y* OR *Z* AND NOT *W*" that specifies the environment of an atom. We assume this reference atom is contained in fragments localized in the set *A* of the global tree. In the first iteration, the atomic environment specification *X* is detected utilizing the local search on a modified fragment space and the search results in the blue tree. The white and the blue trees are superimposed. The inclusion of the first environment *X* (blue) in the global tree is realized by intersecting the fragment sets *A* and *D*. The resulting set stores fragments that simultaneously contain the global SP and the atomic environment SP of *X*. The fragments stored in *E* are attached to the global tree, to complete the atomic environment *X*.

In the following two iterations, the resulting intermediate tree (gray) is modified to include the atomic environment *Y* (turquoise and yellow). Since the local search for the environment *Y* gains two recombination trees, the intermediate tree is duplicated and the copies are extended to include the associated atomic environment in the second iteration: the first turquoise environment is present in a subset of fragments contained in the intermediate tree; therefore, the corresponding sets of fragments are intersected. The second turquoise environment is partially contained in the set $A \cap D$ and *E*. These sets are intersected, and the fragments in set *K* are attached. The two resulting green trees describe products that obey the complete logical specification, since the second part of the logical environment specification is ORed.

The third iteration processes the second part of the logical OR term. The global recombination tree is duplicated and the atomic environment *Z* represented by the purple tree is included. The atomic environment *W* is negated; accordingly the fourth iteration detects the combinations of fragments that contain the atomic environment (red tree) and generates the trees that exclude the respective environment. The result of the exclusion is a set of four trees. In the first of these trees, the atomic environment is excluded by removing the fragments in

dx.doi.org/10.1021/ci400107k | J. Chem. Inf. Model. 2013, 53, 1676−1688

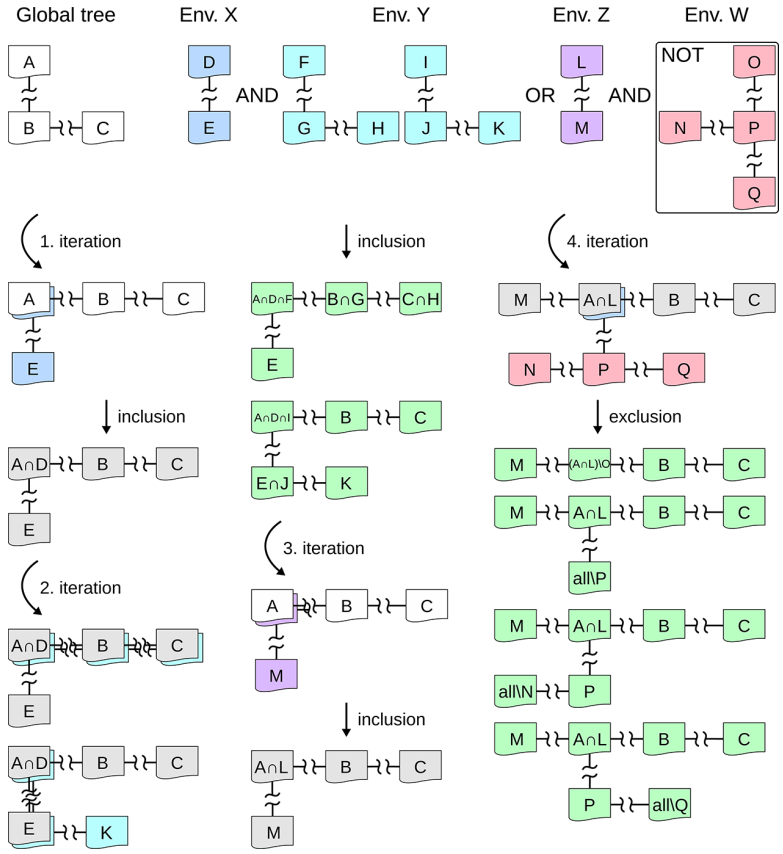


Figure 4. Example showing four iterations of the atomic environment search to include an atomic environment description of the form "*X* AND *Y* OR *Z* AND NOT *W*". The white tree shows the global recombination tree. The logical parts of the atomic environment descriptions are color-coded in blue, turquoise, purple, and red (top). Intermediate trees are gray. Combined trees that represent the result of the four iterations are shown in green. The letters *A* to *Q* represent the sets of fragments contained in nodes of recombination trees. Expressions such as $A \cap D$ indicate the set resulting from intersecting the fragment sets *A* and *D*. Fragment sets such as *all**P* hold all fragments of the fragment space compatible at the current linker without fragments of corresponding set *P*.

set *O* from the fragments in set $A \cap L$. Therefore, any fragments can be attached at fragments of set $(A \cap L)\backslash O$ and the atomic environment *W* will not emerge. In the second exclusion tree, the inversion of set *P* is attached, which prohibits the formation of atomic environment *W*. In the last two exclusion trees, the modification attaches the path to tree nodes and stores the inversion of sets *N* and *Q*. All green trees in Figure 4 describe combinations of fragments that follow the complete logical environment specification and depict the result of processing the atomic environment for a reference atom. If the overall query pattern includes additional atomic environments, these are processed in subsequent iterations similar to the described example.

Enumeration of Products. In the final step of a pattern search, the resulting recombination trees are converted into the set of products containing the search pattern. A product is a fragment or a molecule constructed by connecting fragments. A recombination tree describes such a combination of fragments and holds a list of fragments in each node. Moreover, each node stores possible assignments of the SP to fragments. Since an SP may be assigned to the same fragment in different ways which would result in the enumeration of the same product, the enumeration procedure discards such assignments and removes duplicate fragments with regard to their orientation. Afterward, nodes store only unique sets of fragments. For the enumeration of products from each recombination tree, the algorithm subsequently selects one fragment from every node and connects these fragments according to the recombination tree topology. Even though the fragment sets are unique during that selection step, the combination of different fragments might still lead to the same product, for example, fragments selected from different recombination trees. In order to detect equal products and to circumvent memory limitations, the products are stored in a persistent database using unique SMILES as database keys. A single product represents the smallest fragment composition that contains the pattern of interest. Such a product may still contain open valences that allow a further attachment of fragments or that can be saturated with hydrogen atoms to form a complete molecule.

■ DATA SETS

We evaluated our search method in three use-case examples and additionally in a large scale experiment containing 738 SMARTS patterns.[45] All four experiments were conducted on three different FS's. The breaking of retrosynthetically interesting chemical substructures (BRICS)[21] follows the RECAP[19] approach and comprises 16 types of link atoms and 64 connection rules. The number of fragments varies from 4800 for BRICS 4k to 22000 for BRICS 20k. Even though both FS's contain a relatively small number of fragments, link types,

dx.doi.org/10.1021/ci400107k | J. Chem. Inf. Model. 2013, 53, 1676−1688

and connection rules, the number of products that can be created from theses spaces is arbitrarily large. A general measure for an FS is the number of products that can be created using up to five fragments. BRICS 4k and BRICS 20k contain 10^{16} and 10^{19} of such products, respectively. The KnowledgeSpace is a combination of 82 chemical synthesis protocols and contains 10876 fragments, 488 link types, and 7130 connection rules. Due to its chemical source, the KnowledgeSpace covers about 1.2×10^{10} products that are presumably synthetically accessible.

RESULTS

The following experiments reflect drug development scenarios in which a chemical pattern search is of central interest. In each experiment, FS's are searched for molecules that include a user-defined query pattern. We automatically verified that only molecules including the query were retrieved. The tests reveal insides on the algorithmic run time of our search method and show its general applicability in drug development.

The measurements are intended to reveal dependencies on the number of products contained in an FS and evaluate the search and enumeration times on a single Intel(R) Xeon(R) CPU E5630 2.53 GHz core. The recorded times only comprise the search and the enumeration but exclude preprocessing, e.g., molecule and pattern initialization, and I/O times.

Exploration of Chemical Homologues. The search for molecules that follow specific structural constraints is often applied after a molecular core or chemical class of interest has been experimentally identified. Since FS's contain many molecules that are present neither in vendor catalogs nor in in-house databases, a subsequent search can reveal novel molecules.

We designed our first use-case example to reflect such a screening and queried three different FS's for the presence of two common sulfonamides, sulfachlorpyridazine and sulfachlorpyrazine, as depicted in Figure 5. An initial exact search

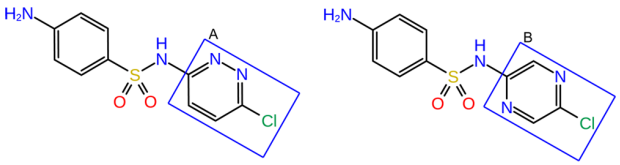


Figure 5. Structure of sulfachlorpyridazine and sulfachlorpyrazine. The ring systems used in the SMARTS query are indicated with A and B.

revealed that both molecules are not contained in either of the three spaces. Nevertheless, the FS's might encode homologue compounds of interest. Therefore, we refined the query as shown in Figure 6 to describe sulfonamides with specific ring systems as present in sulfachlorpyridazine and sulfachlorpyrazine and queried the FS's again. Table 1 summarizes the number of products obtained from the three searches. The two BRICS spaces contain a small number of such sulfonamides, all composed of two fragments joined at the bond between sulfur and nitrogen of the sulfonamide group. This connection is directly defined in the BRICS connection rules. Many of these molecules are similar to sulfachlorpyridazine and sulfachlorpyrazine. Figure 7 shows examples of the extracted molecules and products. The products still include open valences which allow for the further attachment of fragments to obtain even larger molecules and to forward them to lead optimization.

Search times ranged from 3.8 to 47.9 s and enumeration times from 0.08 to 0.7 s. With regard to the large number of possible products contained in each FS, 2.6×10^6 and 1.7×10^8 products with up to two fragments in BRICS 4k and BRICS 20K, respectively, the search times are below 1.5 ms for each product. (The number of possible products with n fragments was numerically calculated with respect to the link types and connection rules. It describes all theoretically possible combinations of up to n fragments, taking into account the connection topology when more than three fragments are connected.) Conventional database searches need around 0.04 ms per molecule,[45] which is in comparison to our search times about 27 times slower. However, a search only supplies a combination of fragments and the corresponding connection of link atoms. If explicit products are required, the relative search times per product moreover include the enumeration and then range around 1.6 ms per product. Nevertheless, an FS in combination with the described search procedure enables a fast pattern search of an arbitrarily large number of molecules in a reasonable time.

Detection of Undesired Products. Since FS's cover a large number of molecules, they might also include compounds that are inappropriate for drug development. They might have undesired physicochemical or structural properties, for example, contain reactive functional groups. Figure 8 shows a pattern that describes skin toxic molecules.[48] A pattern search revealed that both BRICS spaces in fact contain a large number of skin toxic products while the KnowledgeSpace only contains 29 such molecules (Figure 9). Table 2 shows an overview of the search results. The composition of the products with up to five fragments retrieved from the BRICS spaces is due to the generic pattern definition that constructs products involving multiple connection rules. The KnowledgeSpace is much more restricted on how fragments can be connected, and thus, the query leads to a significantly smaller number of skin toxic products. In addition, the KnowledgeSpace originates from chemical synthesis protocols in which toxic products are rather unlikely.

The search times are in accordance with the first experiment and range from 2.0 s to 1.3 min. At the first sight, the enumeration times, especially in BRICS 20k, are surprisingly high with about 6.3 days for 2.3×10^8 products. However, considering that the method enumerated over 200 million products, the average enumeration time per product is 2.3 ms. Therefore, the enumeration time is 1.4 times higher compared to the first experiment, which is a result of the additional demands on the persistent product database when millions of products need to be compared.

Fragment spaces are often used to supply new directions in a drug discovery process. Obviously, only molecules free of reactive groups are suited for lead optimization, and therefore, the number of reactive molecules should be as small as possible in a utilized FS. Our method allows a quantification of products contained in an FS's that include user-defined reactive groups. Moreover, the results in the form of recombination trees obtained from such a search reveal how such reactive products are composed. This information can be used to optimize the FS by modifying the connection rules to reduce the formation of toxic products.

Extraction of Macromolecules under Structural Constraints. Besides small molecules, FS's contain a number of large macromolecules such as oligopeptides. Oligopeptides are used as inhibitors for protein targets such as kinases,

dx.doi.org/10.1021/ci400107k | J. Chem. Inf. Model. 2013, 53, 1676−1688

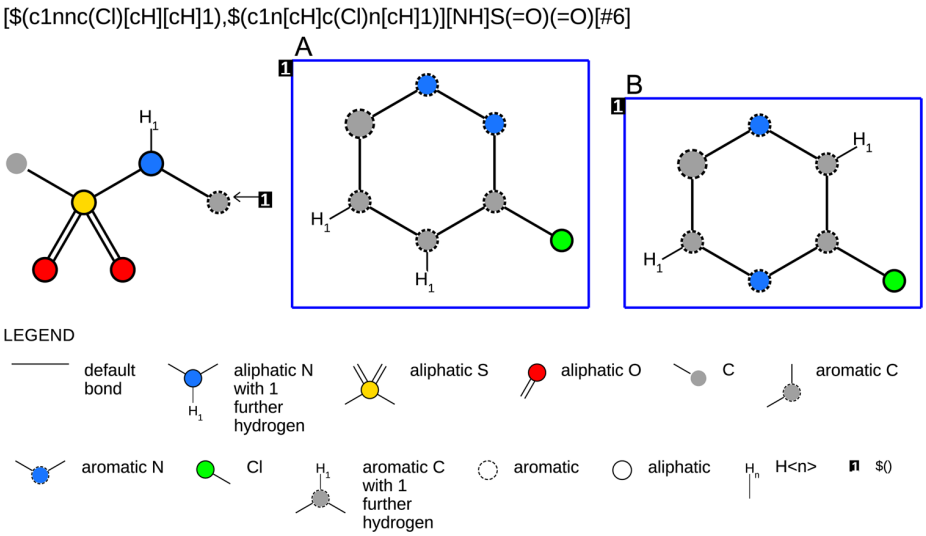


Figure 6. Sulfonamide pattern with two different ring systems defined as recursive atomic environment. The sulfachlorpyridazine ring system is marked with A and the sulfachlorpyrazine with B. The SMARTS string is given in the Supporting Information as "sulfonamides". Figure created with SmartsViewer.[47]

Table 1. Results Querying for Sulfonamides with Restricted Ring Systems

	search time (s)	enum. time (s)	products	
			1 fragment	2 fragments
BRICS 4k	3.84	0.08	0	49
BRICS 20k	47.90	0.70	0	446
KnowledgeSpace	17.81	0.00	0	0

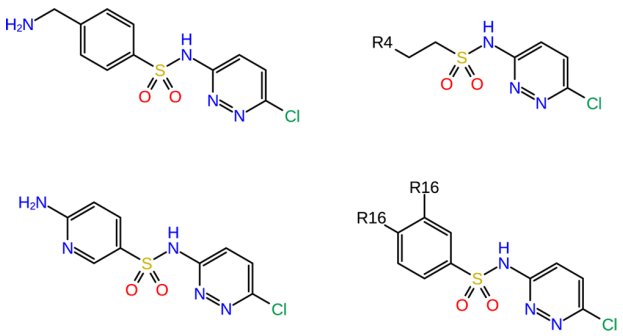


Figure 7. Examples of sulfonamide molecules (left) and products (right) retrieved from BRICS 4k (top) and BRICS 20k (bottom).

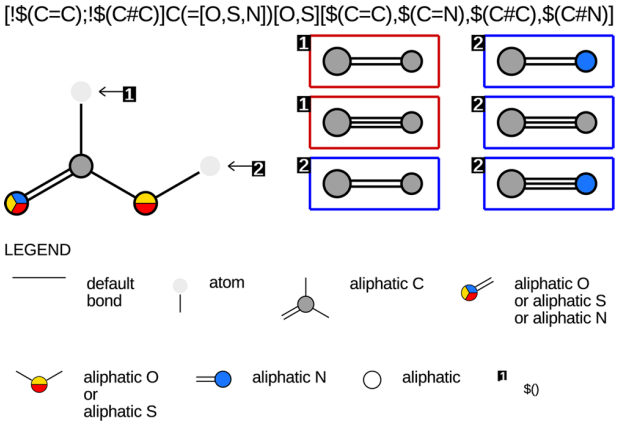


Figure 8. A SMARTS pattern describing skin toxic compounds.[48] All red atomic environments are forbidden at position 1. At least one of the blue environments must be present at position 2 in the retrieved products. The SMARTS string is given in the Supporting Information as "skin toxic". Figure created with SmartsViewer.[47]

proteases, and HIV-1 assembly.[49] In order to show the ability of our method to handle large query patterns including a number of atomic environments, we searched each FS for the occurrence of tripeptides. The search was restricted to identify only tripeptides with specific hydrophobic side chains. Figure 10 depicts the associated query pattern. Table 3 and Figure 11 show that all three FS's contain such tripeptides: BRICS 4k allows the construction of 6 unique peptides. As expected from the higher number of fragments contained in BRICS 20k, the number of found tripeptides rose to 88 products. The largest number of tripeptides was identified in the KnowledgeSpace with 256 molecules. The search times ranged between 45.58 s and 19.03 min. Even though the number of retrieved products was relatively small, the method indirectly considers all possible products with up to five fragments, i.e., 2.3×10^{16} in BRICS 4k, 1.3×10^{19} in BRICS 20k, and 1.2×10^{10} products in the KnowledgeSpace. Assuming that the search would be performed in a fully enumerated space, a single product of the FS was scanned in 1.98 fs in BRICS 4k, 0.09 fs in BRICS 20k, and 82.6 ns in the KnowledgeSpace.

Run Time Statistics. Besides the selected use-case examples, we extensively evaluated the run time of our algorithm with 738 additional searches on all three FS's. The employed SMARTS patterns comprise 504 patterns without and 234 patterns with atomic environment definitions.[45] An overview of the search times in BRICS 4k, BRICS 20k, and KnowledgeSpace is presented in Figure 12. Hydrogens are implicitly contained in the patterns. The histograms show that the algorithm finished 95% of the queries without atomic environments in below 7 s in BRICS 4k, below 300 s in BRICS 20k, and below 200 s in KnowledgeSpace. The median search times for a single query ranged between 0.06 and 3.2 s and the

dx.doi.org/10.1021/ci400107k | J. Chem. Inf. Model. 2013, 53, 1676−1688

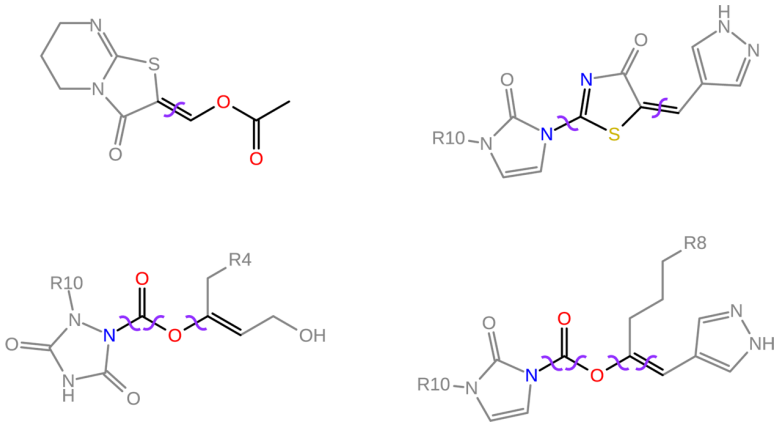


Figure 9. Examples of skin toxic molecules and products retrieved from BRICS 20k. Parts matching the pattern are in color. Former fragment boarders are indicated by purple lines.

Table 2. Results of a Query Describing Skin Toxic Compounds

			products		
	search time (s)	enum. time (h)	1 fragment	2−3 fragments	4−5 fragments
BRICS 4k	2.07	0.09	1	9.7×10^4	1.2×10^5
BRICS 20k	77.61	150.40	21	5.0×10^7	1.8×10^8
KnowledgeSpace	4.03	0.05	5	24	0

maximum between 8.7 and 27.9 min. The time needed to retrieve the desired products drastically rose when atomic environments were defined in the patterns. In these cases, 95% of the queries were finished in 10 min in BRICS 4k and in 2.8 h in BRICS 20k and 8.3 h in KnowledgeSpace. Four instances were aborted after the time limit of 48 h of computational time was reached in BRICS 20k and KnowledgeSpace. These instances were excluded from the median and maximum search time calculations. Median search times for a single query ranged between 8.3 s and 3.6 min, and the maximum ranged between 12.4 and 21.2 h. The SMARTS expressions of the patterns that exceeded the time limit are given in the Supporting Information as Smarts1, Smarts2, Smarts3, and Smarts4. A detailed inspection of Smarts1 and Smarts2 showed that these patterns are of exceptional size and include an outstanding number of atomic environments. In general, both features lead to a dramatic increase in search time. The recursive definitions in Smarts3 of the nitrogen nodes are unfortunate for the algorithm, since they result in a lot of matches that all had to be inspected. We rewrote the pattern (Smarts3′) with an explicit specification of nitrogen nodes which drastically reduced the search time. The pattern was then found in 21.5 min and in 1.2 h in BRICS 20k and KnowledgeSpace, respectively. Smarts4 defines a small symmetric pattern with equal atomic environment definitions in both nodes. The atomic environments are quite generic and lead to over a billion matches that have to be combined into a global match. As a result, the run time exceeds the limit of 2 days.

Obviously, the search times strongly differ between BRICS 4k and the other two FS's. This result is not surprising since BRICS 20k includes about five times more fragments and allows the construction of 100 times more products. All of these products had to be indirectly considered during the search. KnowledgeSpace contains fewer products than BRICS 4k but uses about 30 times more link types and over 100 times more connection rules. The results of KnowledgeSpace led to the conclusion that not only the number of possible products influences the run time but also the complexity of the FS in terms of the number of link types and connection rules.

CONCLUSIONS

The presented algorithm allows an exact search of chemical patterns in nonenumerated chemical spaces. The method supports generic queries with a recursive definition of atomic environments. These atomic environment specifications are sometimes essential for a clear definition of desired chemical patterns. The algorithm identifies the smallest combinations of fragments in an FS for which the corresponding products include the query pattern and obey the atomic environment definitions. The conducted experiments show the applicability of our method to explore chemical homologous, to sample an FS for the occurrences of toxic molecules, and to search for macromolecular structures under structural constraints. In general, large chemical spaces can be searched to identify molecules with desired structural features in reasonable times. However, the search times depend on the complexity of the patterns in terms of size and the number of atomic environments. The complexity of the searched FS with regard to the covered chemical space, the number of link types and connection rules, and the number of products that follow the query pattern has an impact on the search times, as well. In rare cases, the search times exceed a critical limit. As shown in the Supporting Information, such queries are complex and generic and include a large number of atomic environments. However, they are quite rare in practice.

At the current stage, the algorithm enforces a noncyclic connection of fragments to products which is in agreement with most FS processing algorithms and currently available FS's. Nevertheless, in special cases the formation of macromolecular cycles might be wishful. Stereoisomers are currently not addressed by our algorithm. In general, the assignment of a stereocenter is not possible during the connection of fragments,

dx.doi.org/10.1021/ci400107k | J. Chem. Inf. Model. 2013, 53, 1676−1688

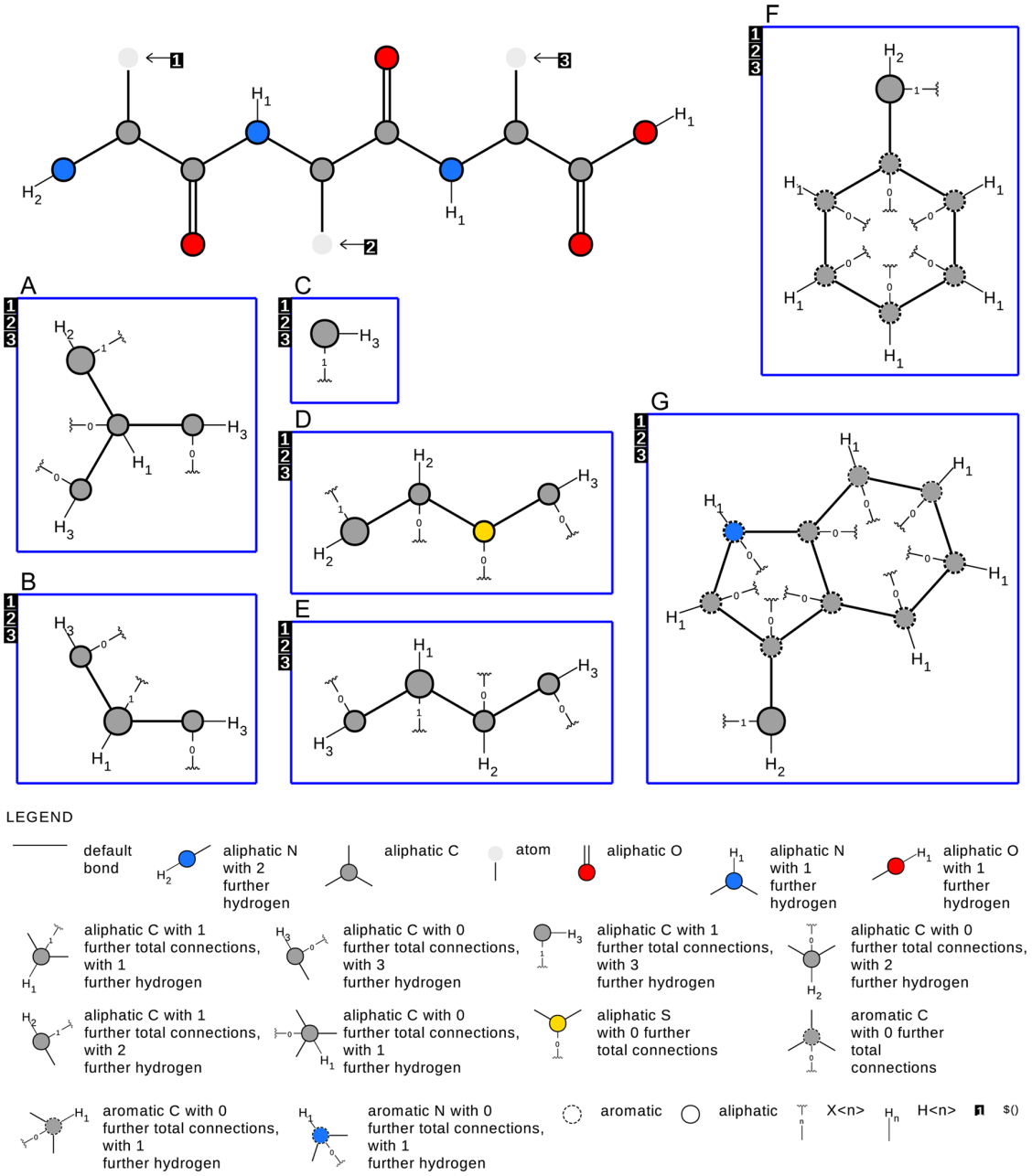


Figure 10. Tripeptide backbone pattern with the restriction to hydrophobic side chains. Atomic environment: (A) leucine; (B) valine; (C) alanine; (D) methionine; (E) isoleucine; (F) phenylalanine; (G) tryptophan. The SMARTS string is given in the Supporting Information as "oligopeptides". Figure created with SmartsViewer.[47]

Table 3. Results of Querying for Tripeptides with Hydrophobic Side Chains

	search time (m)	enum time (s)	products		
			1 fragment	2–3 fragments	4–5 fragments
BRICS 4k	0.76	0.08	0	6	0
BRICS 20k	19.03	0.16	0	50	38
KnowledgeSpace	16.51	0.34	0	256	0

since an additional fragment can create a molecular symmetry that causes stereocenters to vanish. The assignment of stereocenters is only possible on the basis of molecules. Since the search method primarily supplies products, a final correspondence between query and molecule stereocenters can only be part of a postprocessing step.

The options to further process the search results are manifold. Products retrieved from a search provide open attachment points for a follow-up lead optimization. For direct use of the products, the attachment points can be saturated with hydrogen atoms to obtain explicit molecules. Moreover, the intermediate result in the form of a set of recombination trees is a valuable source to modify the corresponding FS. The recombination trees represent the connection patterns of fragments that lead to products following the query. With this information it is possible to modify the connection rules and,

dx.doi.org/10.1021/ci400107k | J. Chem. Inf. Model. 2013, 53, 1676–1688

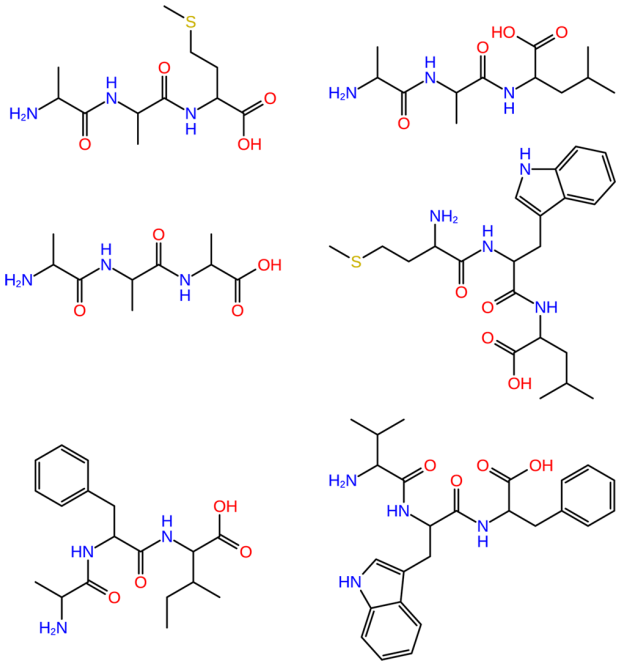


Figure 11. Examples of tripeptides retrieved from BRICS 4k (top), BRICS 20k (middle), and KnowledgeSpace (bottom).

thus, to focus or extend the underlying FS. The recombination tree might even allow the detection of common products encoded by multiple FS's.

■ ASSOCIATED CONTENT

Supporting Information

SMARTS strings referred to in the Experimental section. This material is available free of charge via the Internet at http://pubs.acs.org.

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: rarey@zbh.uni-hamburg.de.

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

We thank Florian Lauck and Melanie Geringhoff for revising the manuscript.

■ REFERENCES

(1) Sussenguth, E. H. A Graph-Theoretic Algorithm for Matching Chemical Structures. *J. Graph Theor.* **1965**, *5*, 36−43.

(2) Figueras, J. Substructure Search by Set Reduction. *J. Graph Theor.* **1972**, *12*, 237−244.

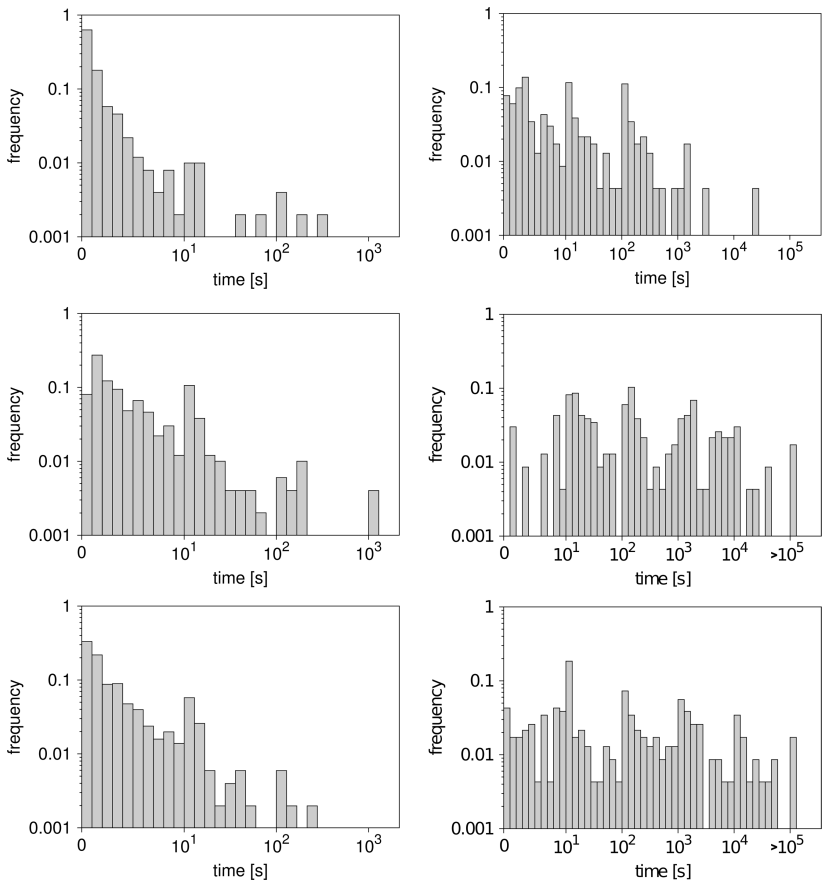


Figure 12. Search time histogram for 504 SMARTS pattern without (left) and 234 with (right) atomic environment definitions in BRICS 4k (top), BRICS 20k (middle), and KnowledgeSpace (bottom). The histogram scale is double logarithmic, the relative frequency is with regard to the number of patterns, and the times are given in seconds (s).

(3) Read, R. C.; Corneil, D. G. The graph isomorphism disease. *J. Graph Theor.* **1977**, *1*, 339−363.
(4) Gati, G. Further annotated bibliography on the isomorphism disease. *J. Graph Theor.* **1979**, *3*, 95−109.
(5) Ullmann, J. R. An algorithm for subgraph isomorphism. *J. Assoc. Comput. Mach.* **1976**, *23*, 31−42.
(6) Attias, R. DARC substructure search system: a new approach to chemical information. *J. Chem. Inf. Comput. Sci.* **1983**, *23*, 102−108.
(7) Heyman, J.; Karasinskia, E.; Giles, P. CAS information services for medicinal chemists. *Drug. Inf. J.* **1982**, *16*, 185−190.
(8) Willett, P.; Barnard, J. M.; Downs, G. M. Chemical Similarity Searching. *J. Chem. Inf. Model.* **1998**, *38*, 983−996.
(9) Cordella, L.; Foggia, P.; Sansone, C.; Vento, M. Performance evaluation of the VF graph matching algorithm. *International Conference on Image Analysis and Processing, 1999. Proceedings*; IEEE Computer Society: Washington, DC, 1999; pp 1172−1177.
(10) Cordella, L. P.; Foggia, P.; Sansone, C.; Vento, M. A (sub)graph isomorphism algorithm for matching large graphs. *IEEE Trans. Pattern Anal.* **2004**, *26*, 1367−1372.
(11) Yan, X.; Yu, P. S.; Han, J. Substructure similarity search in graph databases. *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*; ACM: New York, NY, 2005; pp 766−777.
(12) Golovin, A.; Henrick, K. Chemical Substructure Search in SQL. *J. Chem. Inf. Model.* **2009**, *49*, 22−27.
(13) Blum, L. C.; Reymond, J.-L. 970 Million Druglike Small Molecules for Virtual Screening in the Chemical Universe Database GDB-13. *J. Am. Chem. Soc.* **2009**, *131*, 8732−8733.
(14) Irwin, J.; Shoichet, B. ZINC-a free database of commercially available compounds for virtual screening. *J. Chem. Inf. Model.* **2005**, *45*, 177−182.
(15) Bolton, E. E.; Wang, Y.; Thiessen, P. A.; Bryant, S. H. PubChem: Integrated Platform of Small Molecules and Biological Activities. In *Annual Reports in Computational Chemistry*; Wheeler, R. A., Spellmeyer, D. C., Eds.; Elsevier: Amsterdam, The Netherlands, 2008; Vol. 4, Chapter 12, pp 217−241.
(16) Bohacek, R. S.; McMartin, C.; Guida, W. C. The art and practice of structure-based drug design: A molecular modeling perspective. *Med. Res. Rev.* **1996**, *16*, 3−50.
(17) Fink, T.; Bruggesser, H.; Reymond, J.-L. Virtual Exploration of the Small-Molecule Chemical Universe below 160 Da. *Angew. Chem., Int. Ed. Engl.* **2005**, *44*, 1504−1508.
(18) Reymond, J.-L.; van Deursen, R.; Blum, L. C.; Ruddigkeit, L. Chemical space as a source for new drugs. *Med. Chem. Commun.* **2010**, *1*, 30−38.
(19) Lewell, X. Q.; Judd, D. B.; Watson, S. P.; Hann, M. M. RECAP-retrosynthetic combinatorial analysis procedure: a powerful new technique for identifying privileged molecular fragments with useful applications in combinatorial chemistry. *J. Chem. Inf. Comput. Sci.* **1998**, *38*, 511−522.
(20) Boehm, M.; Wu, T.-Y.; Claussen, H.; Lemmen, C. Similarity Searching and Scaffold Hopping in Synthetically Accessible Combinatorial Chemistry Spaces. *J. Med. Chem.* **2008**, *51*, 2468−2480.
(21) Degen, J.; Wegscheid-Gerlach, C.; Zaliani, A.; Rarey, M. On the art of compiling and using 'drug-like' chemical fragment spaces. *ChemMedChem* **2008**, *3*, 1503−1507.
(22) Lessel, U.; Wellenzohn, B.; Lilienthal, M.; Claussen, H. Searching Fragment Spaces with feature trees. *J. Chem. Inf. Model.* **2009**, *49*, 270−279.
(23) Markush, E. Patent US 1506316.
(24) Simmons, E. S. Markush structure searching over the years. *World Pat. Inf.* **2003**, *25*, 195−202.
(25) Downs, G. M.; Barnard, J. M. Chemical patent information systems. *Wiley Interdiscip. Rev.: Comput. Mol. Sci.* **2011**, *1*, 727−741.
(26) Cosgrove, D. A.; Green, K. M.; Leach, A. G.; Poirrette, A.; Winter, J. A system for encoding and searching Markush structures. *J. Chem. Inf. Model.* **2012**, *52*, 1936−1947.
(27) Gillet, V. J.; Downs, G. M.; Ling, A.; Lynch, M. F.; Venkataram, P.; Wood, J. V.; Dethlefsen, W. Computer storage and retrieval of generic chemical structures in patents. 8. Reduced chemical graphs and their applications in generic chemical structure retrieval. *J. Chem. Inf. Comput. Sci.* **1987**, *27*, 126−137.
(28) Fisanick, W. The Chemical Abstract's Service generic chemical (Markush) structure storage and retrieval capability. 1. Basic concepts. *J. Chem. Inf. Comput. Sci.* **1990**, *30*, 145−154.
(29) Holliday, J. D.; Lynch, M. F. Computer storage and retrieval of generic chemical structures in patents. 16. The Refined Search: An Algorithm for Matching Components of Generic Chemical Structures at the Atom-Bond Level. *J. Chem. Inf. Comput. Sci.* **1995**, *35*, 1−7.
(30) Benichou, P.; Klimczak, C.; Borne, P. Handling Genericity in Chemical Structures Using the Markush Darc Software. *J. Chem. Inf. Comput. Sci.* **1997**, *37*, 43−53.
(31) Rarey, M.; Stahl, M. Similarity searching in large combinatorial chemistry spaces. *J. Comput.-Aided Mol. Des.* **2001**, *15*, 497−520.
(32) Schneider, G.; Neidhart, W.; Giller, T.; Schmid, G. "Scaffold-Hopping" by Topological Pharmacophore Search: A Contribution to Virtual Screening. *Angew. Chem., Int. Ed. Engl.* **1999**, *38*, 2894−2896.
(33) Schneider, G.; Clément-Chomienne, O.; Hilfiger, L.; Schneider, P.; Kirsch, S.; Böhm, H.-J.; Neidhart, W. Virtual Screening for Bioactive Molecules by Evolutionary De Novo Design Special thanks to Neil R. Taylor for his help in preparation of the manuscript. *Angew. Chem., Int. Ed. Engl.* **2000**, *39*, 4130−4133.
(34) Hartenfeller, M.; Proschak, E.; Schüller, A.; Schneider, G. Concept of combinatorial de novo design of drug-like molecules by particle swarm optimization. *Chem. Biol. Drug. Des.* **2008**, *72*, 16−26.
(35) Lippert, T.; Schulz-Gasch, T.; Roche, O.; Guba, W.; Rarey, M. De novo design by pharmacophore-based searches in fragment spaces. *J. Comput.-Aided Mol. Des.* **2011**, *25*, 931−945.
(36) Hartenfeller, M.; Zettl, H.; Walter, M.; Rupp, M.; Reisen, F.; Proschak, E.; Weggen, S.; Stark, H.; Schneider, G. DOGS: Reaction-Driven *de novo* Design of Bioactive Compounds. *PLoS Comput. Biol.* **2012**, *8*, e1002380.
(37) Good, A. C.; Lewis, R. A. New methodology for profiling combinatorial libraries and screening sets: cleaning up the design process with HARPick. *J. Med. Chem.* **1997**, *40*, 3926−3936.
(38) Gillet, V. J.; Willett, P.; Fleming, P. J.; Green, D. V. S. Designing focused libraries using MoSELECT. *J. Mol. Graphics Modell.* **2002**, *20*, 491−498.
(39) Fischer, J.; Lessel, U.; Rarey, M. LoFT: Similarity-Driven Multiobjective Focused Library Design. *J. Chem. Inf. Model.* **2010**, *50*, 1−21.
(40) Domine, D.; Cedric, M. Method for fast substructure searching in non-enumerated chemical libraries. US Patent Application US 2007/0260583 A1, 2007
(41) Ehrlich, H.-C.; Volkamer, A.; Rarey, M. Searching for Substructures in Fragment Spaces. *J. Chem. Inf. Model.* **2012**, *52*, 3181−3189.
(42) *Daylight Theory Manual*, version 4.9; Daylight Chemical Information Systems Inc.: Aliso Viejo, CA, 2008.
(43) Ash, S.; Cline, M. A.; Homer, R. W.; Hurst, T.; Smith, G. B. SYBYL line notation (SLN): A versatile language for chemical structure representation. *J. Chem. Inf. Comput. Sci.* **1997**, *37*, 71−79.
(44) Proschak, E.; Wegner, J. K.; Schueller, A.; Schneider, G.; Fechner, U. Molecular query language (MQL)—A context-free grammar for substructure matching. *J. Chem. Inf. Model.* **2007**, *47*, 295−301.
(45) Ehrlich, H.-C.; Rarey, M. Systematic benchmark of substructure search in molecular graphs—from Ullmann to VF2. *J. Cheminf.* **2012**, DOI: 10.1186/1758-2946-4-13.
(46) Detering, C.; Claussen, H.; Gastreich, M.; Lemmen, C. KnowledgeSpace—a publicly available virtual chemistry space. *J. Chem. Inf. Model.* **2010**, *2* (Suppl 1), O9.
(47) Schomburg, K.; Ehrlich, H.-C.; Stierand, K.; Rarey, M. From structure diagrams to visual chemical patterns. *J. Chem. Inf. Model.* **2010**, *50*, 1529−1535.
(48) Enoch, S. J.; Madden, J. C.; Cronin, M. T. D. Identification of mechanisms of toxic action for skin sensitisation using a SMARTS pattern based approach. *SAR QSAR Environ. Res.* **2008**, *19*, 555−578.

(49) Owens, R. J.; Tanner, C. C.; Mulligan, M. J.; Srinivas, R. V.; Compans, R. W. Oligopeptide inhibitors of HIV-induced syncytium formation. *AIDS Res. Hum. Retroviruses* **1990**, *6*, 1289−1296.

A4 From structure diagrams to visual chemical patterns

Authors:	Karen Schomburg, Hans-Christian Ehrlich, Katrin Stierand, Matthias Rarey
Type of publication:	Journal article
Reference:	Journal of Chemical Information and Modeling, 50(9): 1529–1535, 2010, doi:10.1021/ci100209a
Status:	Published
Legal:	Reprinted with permission from Karen Schomburg, Hans-Christian Ehrlich, Katrin Stierand, and Matthias Rarey. From structure diagrams to visual chemical patterns. Journal of Chemical Information and Modeling, 50(9):15291535, 2010. Copyright 2010 American Chemical Society.

From Structure Diagrams to Visual Chemical Patterns

Karen Schomburg, Hans-Christian Ehrlich, Katrin Stierand, and Matthias Rarey*

Research Group for Computational Molecular Design, Center for Bioinformatics, University of Hamburg, Bundesstrasse 43, D-20146 Hamburg, Germany

Received May 26, 2010

The intuitive way of chemists to communicate molecules is via two-dimensional structure diagrams. The straightforward visual representations are mostly preferred to the often complicated systematic chemical names. For chemical patterns, however, no comparable visualization standards have evolved so far. Chemical patterns denoting descriptions of chemical features are needed whenever a set of molecules is filtered for certain properties. The currently available representations are constrained to linear molecular pattern languages which are hardly human readable and therefore keep chemists without computational background from systematically formulating patterns. Therefore, we introduce a new visualization concept for chemical patterns. The common standard concept of structure diagrams is extended to account for property descriptions and logic combinations of chemical features in patterns. As a first application of the new concept, we developed the SMARTSviewer, a tool that converts chemical patterns encoded in SMARTS strings to a visual representation. The graphic pattern depiction provides an overview of the specified chemical features, variations, and similarities without needing to decode the often cryptic linear expressions. Taking recent chemical publications from various fields, we demonstrate the wide application range of a graphical chemical pattern language.

INTRODUCTION

Countless applications in various chemical fields depend on representations of molecular patterns. These extremely powerful specifications of a set of chemical features range in complexity from very simple substructure descriptions in analyses of molecular similarity up to elaborate logical combinations of functional groups in drug design approaches. The two-dimensional (2D) depiction of molecules as structure diagrams aids chemists to get a quick estimation on the chemical characteristics of compounds despite apparent complicated names. Structure diagrams are often called as being the language of chemistry. However, no comparable standard strategies for visualization of chemical patterns exist so far. Our aim is to initiate a discussion within the chemical society on the development of such a standard. Here we introduce a new approach to a graphic representation of patterns that is based on the recommendations of the IUPAC for chemical structure diagram drawing.[1] In order to demonstrate the relevance of chemical patterns, we applied our visualization concept to examples from recent chemical publications.[2−4]

The most known and used employment of molecular patterns is database searching, where they are used to filter a set of molecules for compounds related to a query. Starting compounds for organic synthesis with certain functional groups can also be found this way. In drug design, known active ligands for a target are analyzed, and the parts presumed to be responsible for activity are mapped to a molecular pattern for finding new active ligands in a database.[5] Compound filtering is another well-used application of molecular patterns. Unwanted chemical properties, like highly reactive functional groups, are excluded in advance from compound libraries used in high-throughput and virtual screenings[6−8] or are avoided in denovo drug design approaches.[9] In combinatorial chemistry, patterns are used for characterizing bonds of complete molecules that are allowed to be broken.[10] The groups of Lewell et al.[11] and Vieth et al.[12] successfully applied pattern-based rules of breaking bonds for creating fragment libraries. Other applications are the use of patterns as 2D molecular descriptors[13] and for pharamcophore matching.[14] An exotic application was published by Hou et al.,[15] who used patterns for describing functional groups of molecules in the course of predicting their catabolic transformation in microbes. Not to be forgotten is the use of molecular patterns in the form of Markush structures in patents. These structures mostly consist of a generic core with several variable but defined R-group moieties. Markush notations are used to define a set of structures that are covered by the patent.[16] However, most of the above-mentioned methods rely on much more generic pattern representations and depend on chemoinformatic tools and algorithms.

Therefore, the respective patterns are represented by computationally processable molecular pattern languages. The most prominent molecular pattern language is the SMILES arbitrary target specifications (SMARTS) language.[17] It originates from the simplified molecular input line entry system (SMILES),[18] a linear notation of molecules, and is an extension of its concept. In addition to the means needed to describe a complete molecule, atoms and bonds can be further specified with properties. Additionally, all specifications can be connected logically by AND, OR, and NOT. Other molecular pattern languages, including molec-

* Corresponding author. E-mail: rarey@zbh.uni-hamburg.de. Telephone: +49-40-42838-7350.

10.1021/ci100209a
Published on Web 08/26/2010

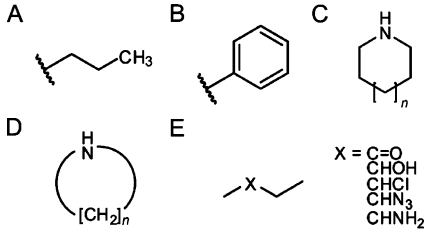


Figure 1. Examples from the IUPAC recommendations on structure diagram drawing for variable structures show graphic display of bonds to unknown moieties (A, B), rings with unknown size (C, D), and a list of variable substituents (E). The figures are adopted from IUPAC recommendations.[1]

ular query language (MQL)[19] and Sybyl line notation (SLN)[20] are built up comparably, differing mainly in the syntax of the language and only slightly in the semantics. In contrast to the traditionally used concept of varying moieties and functional groups at an otherwise fixed compound, these languages provide the means to construct more elaborate chemical feature descriptions.

However, linear notations of molecules and the pattern languages derived from them are designed for effective computational processing. Molecular patterns more closely resemble regular expressions of programming languages than intuitive descriptions of chemical features. As a consequence, the interpretation and the building of a pattern with one of these languages demand a significant learning effort. Due to the syntax of these languages, patterns may get hardly interpretable by humans, even if they are very familiar with the languages. The SMARTS pattern for a sulfonamide group in ionic or neutral form is:

```
[#16;$([#16X4]([NX3])(= [OX1])(= [OX1])[#6]),
      $([#16X4 + 2]([NX3])([OX1 − ])([OX1 − ])[#6])]
```

Although the pattern is not very complicated, the syntax with many brackets leads to a hardly readable expression. Therefore, a more suitable pattern representation is needed. Similar to structure diagrams, a visual depiction of patterns may support scientists in understanding the structural features without much effort. By providing a way to convert this visual depiction to the pattern languages needed for computational methods, the immediate interference of users with these languages can be avoided. The International Union of Pure and Applied Chemistry (IUPAC) recommendations on graphical representations of chemical structure diagrams[1] apply an accepted standard on visualization of compounds in 2D.

However, the recommendations are very limited with respect to the visualization of chemical patterns. Figure 1 presents an assembly of the recommendations concerning variable structures.

Some chemical structure editors offer the possibility of drawing a substructure as a query for database searching.[21−23] The recently developed PubChem chemical structure sketcher can convert such an input to the SMARTS format and also generate a structure out of a SMARTS input string.[23] However, the graphical depiction of query properties is very elementary in a way that the properties are simply written at the concerned atom. For query bonds new symbols with nonobvious denotations are introduced. Another editor that allows the drawing of molecular substructures is the molecular editor Symyx/Draw.[22] Still, the capabilities for generating queries do not cover much of the power of molecular pattern languages. The use of logic operators is not supported for most of the features, and the graphical depiction of query properties is comparable to that of the PubChem sketcher. Since both editors are not capable of depicting recursive SMARTS expressions, only one part of the SMARTS pattern representing sulfonamides (see above) was used as input to both editors (Figure 2).

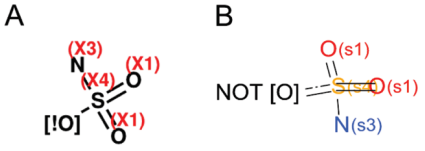


Figure 2. Graphical depiction of the molecular pattern of a sulfonamide (SMARTS: [SX4]([NX3])(=[OX1])(=[OX1])[!O]) generated by the PubChem sketcher[23] (A) and Symyx/Draw[22] (B). The property "total number of connections" (SMARTS: X<*n*>) is printed in both depictions next to the atom element letter. The logic NOT connected to the oxygen is once depicted by an exclamation mark, which is the corresponding SMARTS symbol and once by the text "NOT".

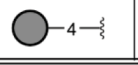
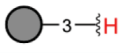
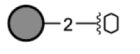
The PubChem sketcher generates the depiction (Figure 2A) automatically, for the depiction by Symyx/Draw (Figure 2B) the pattern has to be drawn manually. Being very alike, both show the limitations of the depiction concept by containing textual overlap with the structure. The depictions are not of great help to scientists not familiar with a pattern language since they still have to figure out details like the meaning of X<*n*> which denotes "the number of total connections of the atom" (depicted by PubChem by the letter "X" next to the atom and by Symyx/Draw by the letter "s").

Due to the lack of existing methods for full depiction of chemical patterns, we developed a visualization concept that is capable of depicting patterns described by chemical features as defined in the SMARTS language. The SMARTS language was chosen as a basis, since it is the most prominent among the linear molecular pattern languages. We considered the following points crucial in developing the visualization concept: First, the effort to learn the new concept has to be significantly less than learning a molecular pattern language itself. Therefore, we based our concept on the well-known representation of molecules as structure diagrams. Second, the depiction should be clear to users without knowledge of molecular pattern languages. Addressing these points, special effort was made to visualize the complete power of the language but to stay consistent with the concept of structure diagrams. Finally, the visualization concept should be generically applicable to different pattern representations. Although the developed visualization concept is based on the SMARTS language, it can be applied—with minor adaptations—to other pattern languages like MQL or SLN, since the semantics and the power of these languages differ only slightly.

In a first application, the concept was realized in a tool that automatically generates a visualization of a SMARTS string named SMARTSviewer available on the Internet (see http://smartsview.zbh.uni-hamburg.de). In the following, first the visualization concept of graphic elements decoding the SMARTS primitives and logic is presented proceeded by a description of the SMARTSviewer implementation. Finally, some exemplary visualizations are discussed.

Table 1. Visualization of Chemical Features That the SMARTS Language Offers to Specify an Atom

Feature	Carbon atom	Aliphatic carbon atom	Aromatic carbon atom	Charge	Mass
SMARTS	[#6]	C	c	[N+1]	[13C]
Color coding					
Element letters					

Feature	Number of total connections	Number of connections to non-hydrogen atoms	Number of connections in a ring
SMARTS	[CX4]	[CD3]	[Cx2]
Color coding			

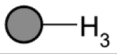
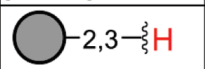
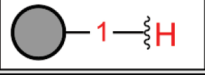
Feature	Number of connected hydrogens	Size of ring	Number of rings	Valency
SMARTS	[CH3]	[Cr5]	[CR1]	[Cv4]
Color coding				

Table 2. Visualization of the Logic Operators OR and NOT in Chemical Patterns[a]

Logic \ Feature		Elements	Number of connections to non-hydrogen atoms	Bonds
OR	SMARTS	[N,C,O]	[C;D2,D3]	C-,=C
	Depiction			
NOT	SMARTS	[!C]	[C!D1]	C!:C
	Depiction			

[a] Red codes the logic NOT. For depiction of the logic OR, the given values are enumerated or the color blue is used.

VISUALIZATION CONCEPT

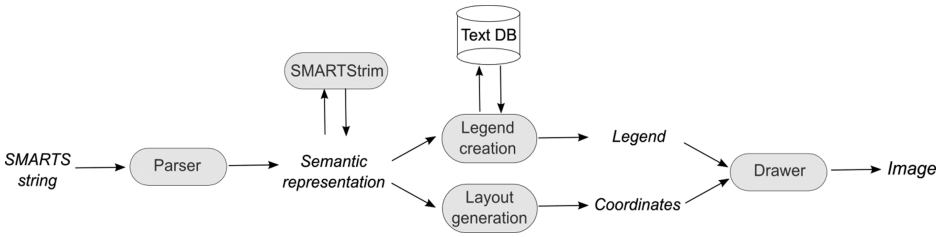
In the SMARTS language, the concept of SMILES specifying the molecule graph with symbols for atoms and bonds is extended by logical operators and several additional symbols specifying properties. Therefore, the visualization concept of structure diagrams has to be extended to cover these additional features in order to achieve a complete visualization. In the concept presented here, the layout of the coordinates of atoms is fully adopted from complete molecules, meaning that, for example, the atoms of a benzene ring are laid out on a hexagon. New graphic symbols are introduced for the depiction of query features. However, all these symbols are based on existing IUPAC recommendations, which are only slightly adapted.

Atoms and Atomic Properties. Additionally to the element, query properties for specifying atoms in SMARTS include features like aromaticity, number of connections, mass, valency, number of connected hydrogens, and several more. Table 1 shows how each of these properties is depicted in the visualization concept. Atoms are drawn as circles that are colored by element. Instead of the color coding, the element can also be depicted by the element letters printed into the circle. In both cases, the stroke of the circle encodes the aromaticity. If none is specified, then no stroke is applied, and if the atom is specified as aliphatic, then a black stroke is applied, and in the case of an aromatic specification, this stroke is dashed. Note that, although aromaticity is a property of a group, most pattern languages consider it as an atom/bond property in order to allow the partial specification of aromatic systems. The value of a given charge is printed at the upper right corner of the atom circle and the mass into the middle of the circle. Several features are available in the SMARTS language to describe the connectivity of an atom. In general, for visualizing this property, a bond ending in a waved line (similar to the IUPAC recommendation for a moiety, see Figure 1A and B) is used. The value of the connectivity given in the SMARTS is printed onto this aborted bond. For the number of connections to nonhydrogen atoms a red colored letter 'H', and for the connections that are in a ring, a hexagon as a symbol for a ring is added behind the waved line. A short bond leading to the letter 'H' subscripted with the value given in the SMARTS indicates the number of connected hydrogen atoms. The features of membership in a ring of a given size and membership in a given number of rings are depicted very similarly with a symbol based on the IUPAC recommendation for variable rings (see Figure 1C and D). While the SMARTS language allows several values for the number of rings an atom participates in (SMARTS primitive 'R<*n*>'), the visualization only covers two cases: "being in a ring" (SMARTS primitive 'R1') and "not being in a ring" (SMARTS primitive 'R0'). Higher values of <*n*> are not recommended since this property is defined over the smallest set of smallest rings (SSSR) base, which is not unique[24] and therefore not considered in our visualization concept. The valency is depicted by small dots drawn inside the atom circle.

Bond Specifications. Bonds are illustrated by one, two and three lines for single, double, and triple bonds, respectively, and by a pair of solid and dashed lines for aromatic bonds, concurring to structure diagram drawing. Wedged bonds are used for depicting tetrahedral stereochemistry (being intrinsically a feature of an atom in the SMARTS language). Additionally, the SMARTS language allows bonds to be specified as a ring or any bond. A bond specified as "any" is depicted by a single line in a light gray color, indicating the unspecified nature of it. For depicting a ring bond, again the symbol of a hexagon for a ring is used and printed onto the bond. Cis and trans configurations around double bonds, being in SMARTS also a feature of bonds, lead to a respective layout of the coordinates of the regarding atoms. The Supporting Information contains a graphic display of all bond types.

Logic Operators. The greatest challenge is the visualization of the SMARTS language's powerful concept of logic. Features describing atoms or bonds can be combined by a logic AND or a logic OR or can be excluded by the logic NOT. In the presented concept, logic is encoded by color. Red indicates the logic NOT and blue the logic OR. Table 2 shows logic combinations of elements, values for the connectivity to non-hydrogen atoms, and bonds. Logical combinations of elements for one atom are depicted by dividing the circle among the elements. Bonds that are connected by OR are placed into a box and colored blue. The logic NOT is depicted by a red circle for elements, a red colored value for connectivities, and a red colored bond. A limitation of the concept is found in the depiction of many elements combined by the logic OR. A division of the circle

Scheme 1. Flowchart of the SMARTSviewer Implementation[a]



[a] An input SMARTS string is parsed into a semantic representation. Before being further processed, the semantic is checked with the SMARTStrim procedure. Then the layout generation provides atom coordinates, and the legend is put together out of a textual database. Both the legend and the coordinates serve as input to a drawer which generates the image.

among more than four elements becomes indistinct. Consequently, if this very rare case occurs, the elements are written as a list in place of the circle (for an example see Supporting Information, Figure 5B). Another special case occurs, if several elements are logically connected to different feature descriptions at one atom. Then an unambiguous assignment of the features to the element is needed, and the features cannot be depicted by their graphic elements. A so-called property bubble is drawn containing the features in form of SMARTS primitives with a connecting line to the respective element fraction (for an example see Supporting Information, Figure 5A).

Recursive Specifications. Recursive expressions, which in SMARTS define the chemical environment of an atom, can be connected by logic operators as well. These specifications are treated as independent molecular graphs and drawn into boxes next to the main molecular graph of the pattern. The color of the stroke of the boxes maps the logic. Consistently, red codes for the logic NOT and blue codes for the logic OR.

Dynamic Legend. The amplification of the visualization concept with a legend serves the purpose to support users in getting accustomed to the new concept. This obstacle is addressed by the first part of the legend, which provides a textual description of each distinct atom and bond. More experienced computational chemists might be familiar with the SMARTS language and therefore only have to get acquainted with the meaning of the graphic symbols. This need is addressed by the second part of the legend, which maps graphic symbols to the respective SMARTS primitives. Both parts of the legend are generated for each SMARTS string individually and consequently contain only information relevant for the depicted pattern.

COMPUTATIONAL METHODS

The presented visualization concept is realized in a first application as a tool that automatically depicts a SMARTS string (see http://smartsview.zbh.uni-hamburg.de). The implementation consists of three main steps which are outlined in Scheme 1. A SMARTS string is processed by a parser which retrieves the semantic information. An interpretation of the semantic is followed by the layout generation and the legend creation. The legend and the coordinates are then further processed by the drawer to generate an image. These steps are described in the following sections.

SMARTS Parsing. In the parsing routine, the input SMARTS string is converted into a tree-like data structure that represents the semantic of the pattern. For this purpose, the SMARTS language is modeled as a context free grammar.[25] The grammar allows to check the correct syntax of the SMARTS string and to extract the contained semantic information. The resulting tree-like representation also contains the precedence of logic operators of the SMARTS language. Therefore, the relevant information, for example, the exact specification of an atom of the pattern, can be extracted easily.

SMARTS Trim. In addition to the syntax test performed by the parser, the semantic is inspected by a procedure called SMARTStrim. This routine is divided into three parts with differing emphases. The SMARTStrim error-check part removes semantic errors. An example is an impossible connection of properties by a logic AND, like an atom that is specified as being a carbon AND a nitrogen. The SMARTStrim simplification part removes redundant information from the SMARTS string, like combinations of property specifications and wildcards. An example is a bond that is specified as "any and ring bonds", which is semantically the same as "ring bond". The third part called SMARTStrim interpretation identifies correlations in the pattern structure and the specifications. An example is an atom that is specified with the property of "being in a ring of size six" but is structurally already in a six-membered ring. All cases which apply to the three parts are listed in the Supporting Information.

Legend Generation. For every atom and bond property which is part of the SMARTS language, a descriptive word or phrase is stored. In the legend creation procedure, the distinct atoms and bonds are identified to avoid redundant legend content. For creating a textual description of an atom or a bond, the respective descriptive words are pieced together with conjunctions concurring with the property and logic specifications of the atom or bond.

Layout Generation. The general layout of a pattern as a chemical graph is consistent with the layout of chemical structure diagrams. Pattern descriptions consist of atoms and bonds as well as any complete molecule, only the specifications of the atoms and bonds differ. Therefore, the assignment of coordinates to the atoms of a pattern is the same problem as that of complete molecules, called a structure diagram generation problem.[26] For every recursive expression in the pattern, a separate chemical graph is created. The relative coordinates are assigned to the atoms with the method of structure diagram generation published by Fricker et al[27] and obey the IUPAC rules of 2D diagram drawing. Afterward, the molecular graphs are placed absolutely with the recursive diagrams next to the main diagram. In addition to structure diagrams, graphic elements that depict specifications are placed at atoms and bonds. In order to support recognition

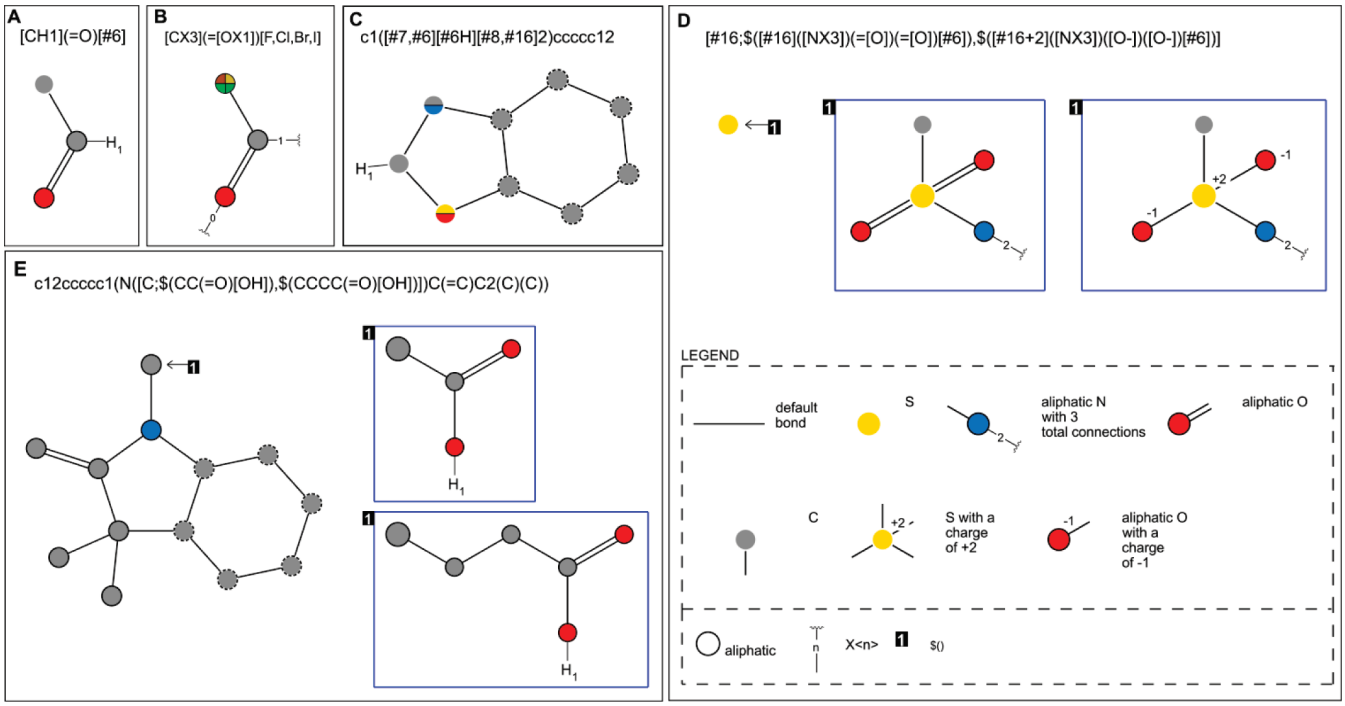


Figure 3. SMARTS expressions visualized by the SMARTSviewer. As the complexity ranges from a pattern describing an aldehyde group (A), via a reactive acyl halide group (B), a pattern matching an aromatic ring system with heteroatoms (C), a sulfonamide group (D), to an indoline group (E) with a chain of variable length to a carboxyl group also the visualization gets more complicated. The visualization of the pattern for a sulfonamide is shown together with the dynamic legend.

of the chemical structure of the pattern, the additional graphic elements are placed without altering the chemical graph layout.

SMARTSviewer Tool. The graphic elements that depict the properties of the pattern are drawn with the open source 2D graphics library cairo.[28] The SMARTSviewer tool can be either used interactively with a Qt[29] based graphical user interface or accessed via a web interface at http://smartsview.zbh.uni-hamburg.de. As input, it takes the pure SMARTS string and generates either pixel- or vector-based images. The user has several possibilities to influence the generated image, among others choosing between color or element letter coding of elements or showing/hiding the legend.

Test Data. For testing the automated generation of the visualization, SMARTS strings that are employed in real applications are used. The SMARTS strings are collected from publications[5,7−9,13,30,31] and the daylight webpage;[32] however, only SMARTS strings that are conform to the Daylight SMARTS specifications are included in the test set. The resulting test set of 762 SMARTS strings ranges from simple patterns used in real applications to highly complicated patterns with many recursions found in the examples from the daylight webpage. The string length varies from 2 to 1008 characters, 247 strings of the set contain recursions. More details on the particular string sets can be found in the Supporting Information.

For showing the relevance of a visualization concept of chemical pattern beyond the application of visualizing SMARTS, some patterns occurring in recent organic chemistry publications[2−4] are visualized according to the visualization concept.

RESULTS AND DISCUSSION

The visualization concept was successfully applied to all SMARTS strings of the test set. Figure 3 shows the visualization of five exemplary chemical patterns automatically generated with the SMARTSviewer. The first three examples are rather simple patterns, representing an aldehyde functional group (A), an acyl halide group (B), which may be used as a filter for reactive components, and an aromatic ring system with heteroatoms at specified positions (C), which was taken from Vechorkin et al.,[2] where the variable parts of the structure are listed. These three examples demonstrate the depiction of elements and aromaticity, while the second example (B) highlights the handling of connectivities. In the SMARTS string, the carbon atom is specified as having three connections (SMARTS primitive ‘X3’. This feature is depicted by a truncated line with the value of the further connections placed onto it, meaning that the already covered connections are subtracted and only the unsaturated connections remain. In this case, two of the three connections are already covered by the structure. The fourth pattern (D) matches sulfonamide groups, which are the common functional group of antibacterial “sulfa drugs”, either in the ionic or neutral form. Here the visualization of recursive SMARTS expressions in boxes next to the main graph is demonstrated as well as the two parts of the legend. Since the two recursive groups are connected by a logic OR, the borders of the boxes are colored blue. For a better identification of the recursively specified atom in the boxes, this atom is drawn with an enlarged radius. The upper part of the legend describes each distinct atom and bond with a short text, while the lower part maps the SMARTS symbols used in the string to the respective graphic elements. The fifth visualization example (E) shows a pattern for an indoline with a variable chain

c1c[c;$(cc1ccccc1), $(cC(=O)OC),$(c-[Br,Cl,C]), $(cC(F)(F)F), $(cO[C;$([CH3]),$(C[CH3])]),$(c[N;$(N([CH3])[CH3]), $(N(=O)=O)])]cc([#7]=[#6](C[Br,Cl])[#16]2)c21

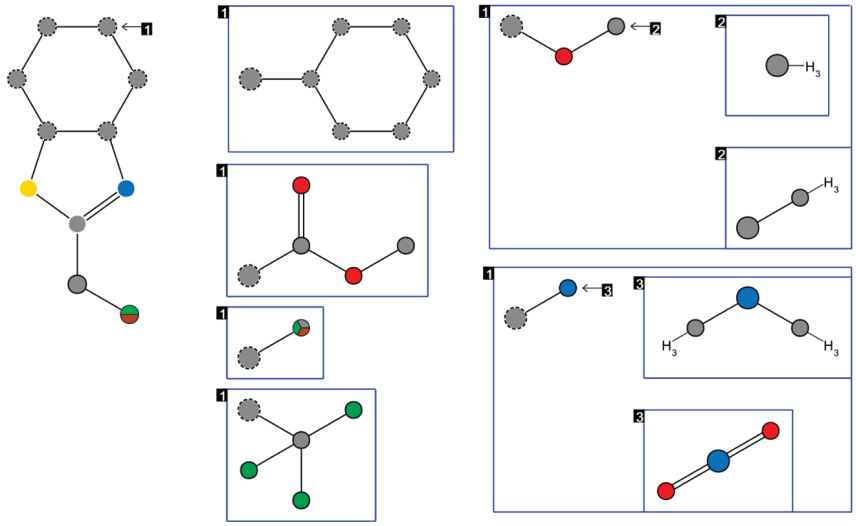


Figure 4. Visual representation of a chemical pattern that represents the starting compound in a synthesis of benzothiazole derivatives active in bacterial cell division inhibition.[4] The compound is substituted with a variable moiety. The recursive expressions show the differences and similarities of the allowed substituents and provide an overview of all possibilities.

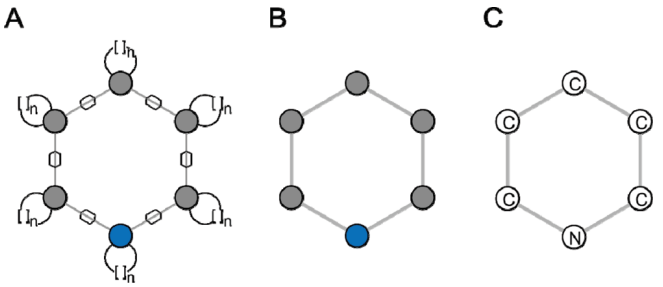


Figure 5. Visualization of the SMARTS pattern [CR]@1@[CR]@[CR]@[CR]@[CR]@[NR]@1 without the SMARTStrim procedure (A), with using the SMARTStrim procedure resulting in a much simpler but with regards to the chemical meaning identical visualization (B) and in the viewing mode of presenting elements not by color but by element letters (C).

length to the carboxyl group. This example was taken from Meguellati et al.,[3] where two structures are drawn, one for each chain length. The depiction in form of a pattern highlights the common and variable parts of the structure.

A more complex pattern is shown in Figure 4. The pattern representing benzothiazole derivatives being used in the synthesis of inhibitors of a bacterial cell division protein was adopted from Haydon et al.,[4] where the various substituents are listed in the form of abbreviations. An advantage of the representation in the form of a visual pattern is the possibility to take in all substituents at a glance instead of looking them up and interpreting the abbreviations. Additionally, the differences and similarities of the variations are emphasized by the depiction.

The effect of the SMARTStrim procedure is highlighted by the example in Figure 5. The visualization of the SMARTS string [CR]@1@[CR]@[CR]@[CR]@[CR]@[NR]@1 without applying the SMARTStrim procedure leads to an unnecessarily complicated figure (A). However, the SMARTStrim procedure recognizes that the pattern can be simplified without changing its meaning. All bonds in this pattern are specified as ring bonds, although the structure of the pattern already satisfies this condition. The same is true for the atoms of the ring, which are specified as "being in a ring". Therefore, the SMARTStrim procedure simplifies the visualization significantly (B). While in all previous figures, elements are depicted by color, Figure 5 C employs the element letter viewing mode. Both depictions convey certain advantages. The color coding supports a very quick recognition of the structure and highlights heteroatoms. The element letter representation is the classic way of depicting elements in 2D structure diagrams and may thus be easier to get acquainted with, since not every scientist may be familiar with the color code. In general, the choice will depend on every scientist's personal preferences, and therefore, both viewing options are realized in the SMARTSviewer. Still, both modes rely on the use of color for depicting logic combinations. In principal, it is possible to create a pure black and white depiction by introducing graphic elements for logic expressions. However, we preferred the color depiction in order to avoid introducing more graphic elements in addition to the ones depicting features.

While the visualization concept could be evaluated concerning the ability of visualizing the complex and powerful SMARTS specifications, the real evaluation will be through the chemical community. Since a visualization concept has to evolve through the needs of science it represents, the visualization concept shown here should be seen as a starting point for the development of a unified chemical pattern language. Current limitations of the concept that may be addressed in future discussions concern extensions of the drawing concept for pattern specifications that are not part of the SMARTS language. Chemical pattern languages, such as SMARTS, are extremely powerful. It is possible to describe highly complex patterns, for example, due to the extensive use of logic expressions. The graphic display of such a pattern is obviously not simple, and, in some cases, probably impossible. Therefore, the visualization concept concentrates on depicting patterns that a scientist may come across in real applications. In our opinion, one of the most important characteristics of a visual pattern language is that

simple patterns result in simple depiction, while more complicated patterns may result in more complicated graphical representations. The examples generated with SMARTSviewer show that the visualization concept presented here achieves this aim in most cases.

CONCLUSION

In summary, we have introduced a new visual representation of chemical patterns. The natural way of communicating structures in the chemical society is via structure diagrams. Therefore, a visual pattern representation must not diverge extensively from the concept of structure diagram drawing. For the full depiction of chemical features that may be used to describe variable structures, this concept had to be extended with several new graphic symbols. For providing an easy way to get acquainted with these, the visualization concept comprises a legend that depicts the meaning of the pattern by a textual description. In a first application, the visualization concept was realized as an automated depiction of the often hardly interpretable SMARTS patterns. This new visual pattern representation has to be seen as a starting point in improving the communication of molecular patterns among scientists. It responds to two problems of handling chemical patterns: First, it offers a general representation of chemical patterns, and second, it improves the usability of computational molecular pattern languages by depicting the linear form graphically.

ADDITIONAL INFORMATION

The SMARTSviewer tool can be accessed freely via the Internet at http://smartsview.zbh.uni-hamburg.de.

Supporting Information Available: The complete visualization concept covering the full power of the SMARTS language and details of the SMARTStrim concept. This material is available free of charge via the Internet at http://pubs.acs.org.

REFERENCES AND NOTES

(1) Brecher, J. Graphical Representation Standards for Chemical Structure Diagrams (IUPAC Recommendations 2008. *Pure Appl. Chem.* **2008**, *80* (2), 277–410.

(2) Vechorkin, O.; Proust, V.; Xile, H. The Nickel/Copper-Catalyzed Direct Alkylation of Heterocyclic C-H Bonds. *Angew. Chem.* **2010**, *122* (17), 3125–3128. *Angew. Chem., Int. Ed. Engl.*, **2010**, *49*(17), 3061−3064.

(3) Meguellati, K.; Koripelly, G.; Ladame, S. DNA-Templated Synthesis of Trimethine Cyanine Dyes: A Versatile Fluorogenic Reaction for Sensing G-Quadruplex Formation. *Angew. Chem.* **2010**, *122* (15), 2798–2802. *Angew. Chem., Int. Ed. Engl.* **2010**, *49*(15), 2738−2742.

(4) Haydon, D. J.; Bennet, J. M.; Brown, D.; Collins, I.; Galbraith, G.; Lancett, P.; Macdonald, R.; Stokes, N. R.; Chauhan, P. K.; Sutariya, J. K.; Nayal, N.; Srivastava, A.; Beanland, J.; Hall, R.; Henstock, V.; Noula, C.; Rockley, C.; Czaplewski, L. Creating an Antibacterial with in Vivo Efficacy: Synthesis and Characterization of Potent Inhibitors of the Bacterial Cell Division Protein FtsZ with Improved Pharmaceutical Properties. *J. Med. Chem.* **2010**, *53* (10), 3927–3936.

(5) Enoch, S. J.; Madden, J. C.; Cronin, M. T. D. Identification of mechanisms of toxic action for skin sensitisation using a SMARTS pattern based approach. *SAR QSAR Environ. Res.* **2008**, *19*, 555–578.

(6) Baurin, N.; Baker, R.; Richardson, C.; Chen, I.; Foloppe, N.; Potter, A.; Jordan, A.; Roughley, S.; Parrat, M.; Greaney, P.; Morley, D.; Hubbard, R. E. Druglike Annotation and Dublicate Analysis of a 23-Supplier Chemical Database Totalling 2.7 Million Compounds. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 643–651.

(7) Walters, W. P.; Murcko, A. M. Prediction of 'drug-likeness'. *Adv. Drug Delivery Rev.* **2002**, *54*, 255–271.

(8) Hann, M.; Hudson, B.; Lewell, X.; Lifely, R.; Miller, L.; Ramsden, N. Strategic Pooling of Compounds for High-Throughput-Screening. *J. Chem. Inf. Comput. Sci.* **1999**, *39* (5), 897–902.

(9) Maass, P.; Schulz-Gasch, T.; Stahl, M.; Rarey, M. Recore: A Fast and Versatile Method for Scaffold Hopping Based on Small Molecule Crystal Structure Conformations. *J. Chem. Inf. Model.* **2007**, *47*, 390–399.

(10) Schneider, G.; Fechner, U. Computer-Based De Novo Design of Drug-Like Moleculs. *Nat. Rev. Drug Discovery* **2005**, *4* (8), 649–63.

(11) Lewell, X. Q.; Judd, D. B.; Watson, S. P.; Hann, M. M. RECAPs Retrosynthetic Combinatorial Analysis Procedure: A Powerful New Technique for Identifying Privileged Molecular Fragments with Useful Applications in Combinatorial Chemistry. *J. Chem. Inf. Comput. Sci.* **1998**, *38* (3), 511–522.

(12) Vieth, M.; Siegel, M. G.; Higgs, R. G.; Watson, I. A.; Robertson, D. H.; Savin, K. A.; Durst, G. L.; Hipskind, P. A. Characteristic Physical Properties and Structural Fragments of Marketed Oral Drugs. *J. Med. Chem.* **2004**, *47* (1), 224–232.

(13) Olah, M.; Bologa, C.; Oprea, T. I. An automated PLS search for biologically relevant QSAR descriptors. *J. Comput.-Aided Mol. Des.* **2004**, *18*, 437–449.

(14) Van Drie, J. H.; Weininger, D.; Martin, Y. C. ALADDIN: An integrated tool for computer-assisted molecular design and pharmacophore recognition from geometric, steric, and substructure searching of three-dimensional molecular structures. *J. Comput.-Aided Mol. Des.* **1989**, *3*, 225–251.

(15) Hou, B. K.; Wackett, L. P.; Ellis, L. B. M. Microbial Pathway Prediction: A Functional Group Approach. *J. Chem. Inf. Comput. Sci.* **2003**, *43* (3), 1051–1057.

(16) Lynch, M. F.; Barnard, J. M.; Welford, S. M. Computer Storage and Retrieval of Generic Chemical Structures in Patents. 1. Introduction and General Strategy. *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 148–150.

(17) *Daylight Theory Manual*, version 4.9; Daylight Chemical Information Systems, Inc.: Aliso Viejo, CA, 2008; http://www.daylight.com/dayhtml/doc/theory/index.html. Accessed July 20, 2010.

(18) Weininger, D. SMILES,a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules. *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 31–36.

(19) Proschak, E.; Wegner, J. K.; Schüller, A.; Schneider, G.; Fechner, U. Molecular Query Language (MQL) - A Context-Free Grammar for Substructure Matching. *J. Chem. Inf. Model.* **2007**, *47*, 295–301.

(20) Homer, R. W.; Swanson, J.; Jilek, R. J.; Hurst, T.; Clark, R. D. SYBYL Line Notation (SLN): A Single Notation To Represent Chemical Structures, Queries, Reactions, and Virtual Libraries. *J. Chem. Inf. Model.* **2008**, *48*, 2294–2307.

(21) Bruno, I. J.; Cole, J. C.; Edgington, P. R.; Kessler, M.; Macrae, C. F.; McCabe, P.; Pearson, J.; Taylor, R. New Software for Searching the Cambridge Structural Database and Visualizing Crystal Structures. *Acta Crystallogr., Sect. B: Struct. Sci.* **2002**, *58*, 389–97.

(22) *Symyx/Draw*, version 3.2; Symyx Technologies, Inc.: Sunnyvale, CA, 2009.

(23) Ihlenfeldt, W. D.; Bolton, E. E.; Bryant, S. H. The PubChem Chemical Structure Sketcher. *J. Cheminf.* **2009**, *1*, 20.

(24) Berger, F.; Flamm, C.; Gleiss, P. M.; Leydold, J.; Stadler, P. F. Counterexamples in Chemical Ring Perception. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 323–331.

(25) Alfred, V. A.; Ullmann, J. D. In *The Theory of Parsing, Translation, and Compiling: Parsing*; Prentice-Hall, Inc.: London, 1972; Vol. 1, Chapter 2, pp138−167.

(26) Helson, H. E. Structure Diagram Generation. In *Reviews in Computational Chemistry*, Wiley-VCH: New York, 1999; Vol. 13, pp 313−398.

(27) Fricker, P.; Gastreich, M.; Rarey, M. Automated Drawing of Structural Molecular Formulas under Constraints. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 1065–1078.

(28) *Cairo Graphics Library*, version 1.8.8; Nokia Corporation: Oslo, Norway; Redwood City, CA, http://www.cairographics.org. Accessed July 20, 2010.

(29) *Qt Application Development Framework*, version 4.6.0; Nokia Qt Development Frameworks: .

(30) Abolmaali, S. F. B.; Wegner, J. K.; Zell, A. The compressed feature matrix - a fast method for feature based substructure search. *J. Mol. Model.* **2003**, *9*, 235–241.

(31) Agrafiotis, D. K.; Gibbs, A. C.; Zhu, F.; Izrailev, S.; Martin, E. Conformational Sampling of Bioactive Molecules: A Comparative Study. *J. Chem. Inf. Model.* **2007**, *47*, 1067–1086.

(32) *Daylight SMARTS examples*; Daylight Chemical Information Systems, Inc.: Laguna Niguel, CA; http://www.daylight.com/dayhtml_tutorials/languages/smarts/smarts_examples.html. Accessed May 25, 2010.

CI100209A

A5 Torsion angle preferences in drug-like chemical space: A comprehensive guide

Authors:	Christin Schärfer, Tanja Schulz-Gasch, Hans-Christian Ehrlich, Wolfgang Guba, Matthias Rarey, Martin Stahl
Type of publication:	Journal article
Reference:	Journal of Medicinal Chemistry, 56(5), 2016–2028, 2013, doi:10.1021/jm3016816
Status:	Published
Legal:	Reprinted with permission from Christin Schrfer, Tanja Schulz-Gasch, Hans-Christian Ehrlich, Wolfgang Guba, Matthias Rarey, and Martin Stahl. Torsion angle preferences in druglike chemical space: A comprehensive guide. Journal of Medicinal Chemistry, 56(5):20162028, 2013. Copyright 2013 American Chemical Society.

Journal of Medicinal Chemistry

Article

pubs.acs.org/jmc

Torsion Angle Preferences in Druglike Chemical Space: A Comprehensive Guide

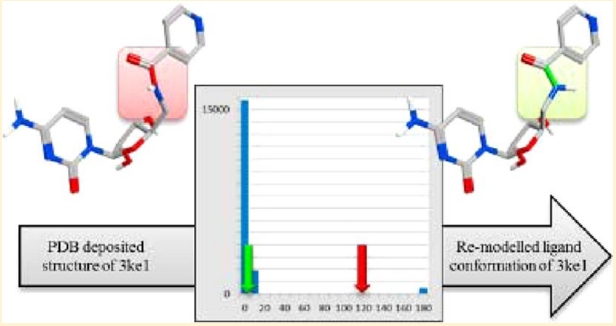
Christin Schärfer,[†] Tanja Schulz-Gasch,*[,‡] Hans-Christian Ehrlich,[†] Wolfgang Guba,[‡] Matthias Rarey,*[,†] and Martin Stahl[‡]

[†]Center for Bioinformatics, University of Hamburg, Bundesstrasse 43, D-20146 Hamburg, Germany

[‡]Discovery Chemistry, F. Hoffmann-La Roche Ltd., CH-4070 Basel, Switzerland

Supporting Information

ABSTRACT: Crystal structure databases offer ample opportunities to derive small molecule conformation preferences, but the derived knowledge is not systematically applied in drug discovery research. We address this gap by a comprehensive and extendable expert system enabling quick assessment of the probability of a given conformation to occur. It is based on a hierarchical system of torsion patterns that cover a large part of druglike chemical space. Each torsion pattern has associated frequency histograms generated from CSD and PDB data and, derived from the histograms, traffic-light rules for frequently observed, rare, and highly unlikely torsion ranges. Structures imported into the corresponding software are annotated according to these rules. We present the concept behind the tree of torsion patterns, the design of an intuitive user interface for the management and usage of the torsion library, and we illustrate how the system helps analyze and understand conformation properties of substructures widely used in medicinal chemistry.



INTRODUCTION

The ability to both understand and generate accessible low-energy conformers is indispensable for small molecule drug discovery. Yet textbook knowledge on small molecule conformations only partly covers druglike chemical space. For structures involving heteroatoms and an interplay of steric and electronic effects, the practitioner is often led to believe that structure generators[1−12] or force field minimizers will somehow provide the right answers. There is a gap between what is in principle known about conformation preferences and the application of this knowledge in day-to-day decision making in medicinal chemistry. Here, we are aiming at giving easy access to this conformational preference information.

To a good first approximation, conformations differ in their torsion angles only. Small rings are usually highly constrained, the acyclic part of the molecule is, however, very flexible and responsible for large conformational spaces. These torsions are frequently independent from each other, allowing stepwise construction processes widely used in methods for the generation of conformers. Many computational approaches to drug discovery are based on sets of diverse low-energy conformations generated by such methods.[13−18] Here we use the same principles for the assessment rather than the generation of conformations. Given a modeled or experimentally determined 3D structure of a small molecule, typical questions are the following: Does it contain strained parts? How far does it deviate from a low-energy conformation? How conformationally flexible is it? To address such questions, a conformer generator is ill-suited, as the combinatorial nature of conformer space creates far more data than necessary.

Information about preferred (=low-energy) regions of individual torsion angles can be obtained from crystal structural data. The growing number of entries in the Cambridge Structural Database (CSD)[19] and the Protein Data Bank (PDB)[20] allow derivation of more and more reliable and specific rules, and efficient tools exist to do this.[21,22] Here we present a torsion library that covers a large part of druglike chemical space. Our approach differs from previous ones[23,24] in that it is comprehensive, yet expert-driven: Instead of automated enumeration of atom sequences and creation of torsion histograms, each step of the process has been subjected to manual curation. Torsion patterns are defined in a hierarchical fashion for successively more complex and specific chemical motives at either end of the rotatable bonds. More specific rules were created whenever a more generic rule would lead to ambiguous or contradictory results and where sufficient data were available. We see this work as an extension of our previous empirical compilations of conformation and interaction preferences.[25−29]

The article is structured as follows: First, we describe the concept of storing torsion patterns and their preferred

Received: November 15, 2012
Published: February 4, 2013

orientations in a hierarchical tree. We briefly present the graphical user interface of a tool named Torsion Analyzer designed to support both the management of the torsion library and its application (technical details on these aspects can be found in the section Materials and Methods). This is followed by an analysis of how the current torsion library covers chemical space relevant for medicinal chemistry. Finally we compare torsion statistics derived from small molecule crystallographic data with those derived from protein ligands and discuss further applications of our methodology.

RESULTS AND DISCUSSION

Torsion Library Concept and Interface. A torsion pattern is a molecular substructure that defines at least the four atoms required to describe a torsion angle (Figure 1). The

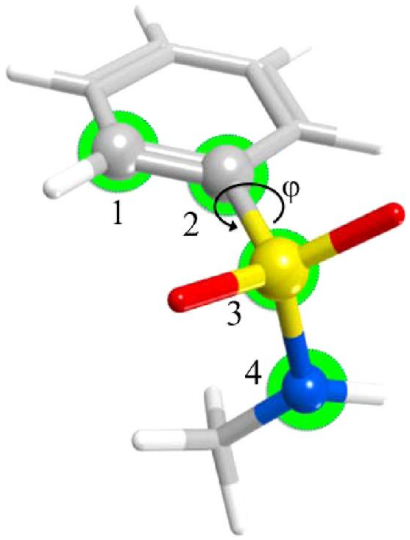


Figure 1. A torsion angle ϕ is defined by four consecutive covalently bonded atoms. It is the angle by which atom 4 has to rotate counterclockwise around the axis defined by the rotatable bond (atoms 2 and 3) to be in plane with atoms 1, 2, and 3. Note that the order of atoms, 1−4 or 4−1, does not affect ϕ.

chemical environment of the rotatable bond may be more precisely defined by encoding further substructure elements linked to this four-atom sequence. We consider each acyclic single bond as a potentially rotatable bond.

The centerpiece of this work is the idea of creating a comprehensive list of such torsion patterns associated with typically observed torsion angle ranges and to organize this list in a hierarchical fashion. This requires choices to be made both regarding the detail of chemical environment to be encoded in the torsion patterns and regarding the type of hierarchical classification. As opposed to unsupervised rule extraction approaches by others,[23,24] we have opted for an expert-driven process in establishing patterns and deriving rules.

At the highest level, our hierarchical system consists of one generic class and six specific classes of torsion patterns. The generic class only defines the two atoms linked by the rotatable bond such that atoms 1 and 4 can be of any type. In addition, the generic class contains one subclass with general rules on rotatable bonds between multiple types of aromatic atoms, independent of element specification. The generic class is used as a fallback option in case no more specific rules have been derived. The other six top-level classes encode more specific patterns for the most important types of rotatable bonds. These are CC, CN, CO, CS, NS, and SS bonds. Each of these classes has multiple subclasses covering named functional groups or substructures. Within these subclasses, torsion patterns are sorted by decreasing specificity such that the more general patterns are found below the more specific ones. As described in the methods section below, this sorting order requires fewer match attempts by the search algorithm.

Each torsion pattern in the library is associated with histograms derived from small molecule crystal structure information contained in the CSD and the PDB databases. Histogram peaks represent the most frequently observed torsion values and correspond to local minima of the torsion distributions. We have used a simple automated process to identify such minima: Histogram bins containing more than 4% of the histogram entries are considered as peak candidates. Two tolerance ranges around these minima were defined as a function of the width of the peaks. This is done by a simple waterline model; inner and outer tolerances reach the bins where occupancy drops to 2.5% and 1.5%, respectively. All automated rules generated in this fashion were visually checked and in many cases slightly adjusted. The rule system we use does not necessarily reflect the position of energetic minima; it merely identifies torsion angles that have been frequently observed. However, the relative energies of conformational states are reflected in the histogram itself, which should serve as an important visual aid in any application of the Torsion Analyzer. It is important to note that being outside the tolerance range is not necessarily a measure of the "quality" of a conformation nor are they an accurate measure of strain energy. They are rough indicators of the likelihood of a specific conformation to be observed in a crystal structure.

Figure 2 is a snapshot of the user interface for managing and applying the torsion library. A panel on the left allows browsing, viewing, and editing of the torsion patterns. Classes can be collapsed and expanded to view subclasses and their content. Torsion patterns can be moved up or down within their class or subclass. Each pattern can be switched on or off, indicated by the little cross to the left of the pattern. Upon selection of a torsion pattern in the tree view, the torsion data associated with this pattern are shown on the right-hand side of the interface. This includes the SMARTS pattern (top), a tabular view of defined minima and tolerances (center), and the corresponding torsion histogram (bottom). Histogram views can be adjusted to display CSD data, PDB data, or both. Histograms either be displayed from −180° to 180° or in condensed form where only the absolute values of torsion angles are regarded. The latter view is sufficient for the current version of the torsion library because of the absence of chiral patterns. Rotatable bonds of molecules loaded into the central 3D viewer are color-coded on the fly by means of a traffic-light coloring scheme. Torsion angles within the first tolerance value around a local minimum are colored green. Those between the first and second tolerance values are orange, and those exceeding the second tolerance are red. Selection of a colored bond in the 3D viewer highlights the four atoms used to measure the torsion angle and the matching SMARTS pattern in the torsion library. On the right, the matching torsion histogram is shown including a marker indicating the value measured at the selected bond.

A first application example illustrating the importance of conformation energy is shown in Figure 3. The two potent

dx.doi.org/10.1021/jm3016816 | J. Med. Chem. 2013, 56, 2016−2028

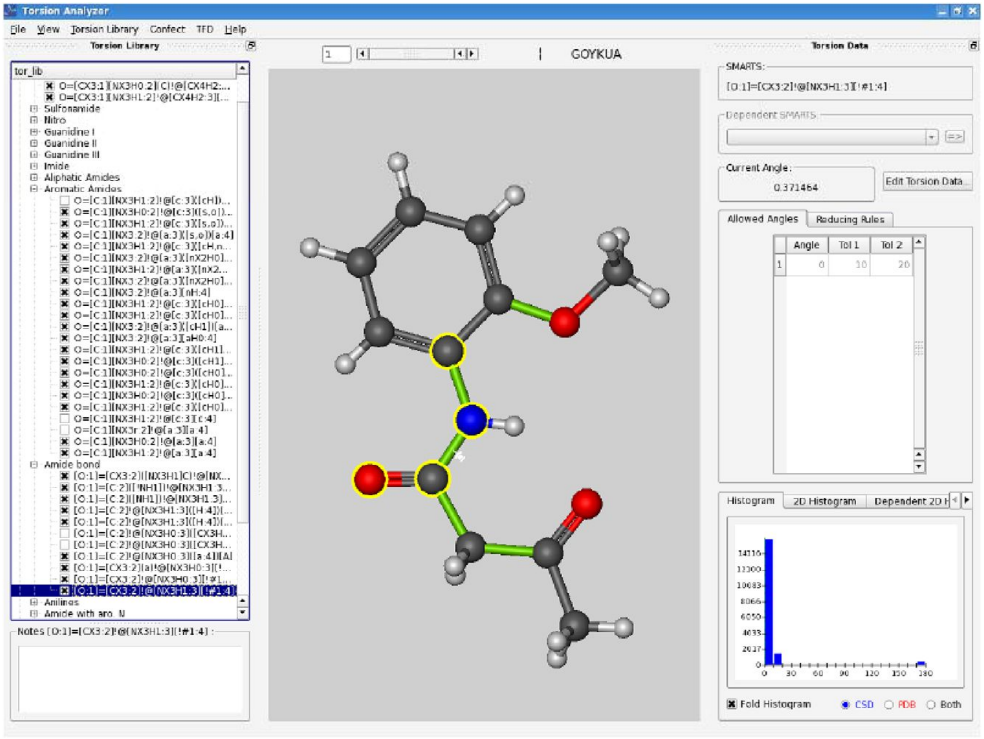


Figure 2. Graphical user interface of the Torsion Analyzer: (left) tree view of the torsion library; (center) 3D panel; (right) assigned minima, tolerances, and torsion histograms.

factor Xa inhibitors,[30] **1** and **2**, differ by one methyl group and a factor of 4 in binding affinity. A look at the binding mode of inhibitor **2** in PDB structure 1ksn[31] reveals that the additional methyl group is solvent exposed and does not form an interaction with the protein. Could differences in conformation preferences contribute to the increase in affinity? The histogram for the torsion angle around the N substituent of the secondary amide shows a single, broad peak at zero degrees. The torsion angle of this bond, which plays a major role in positioning S1 and S4 moieties relative to each other, is therefore in an optimal range. The methyl group introduces a conformational lock; the only accessible rotamer has the hydrogen atom in a syn arrangement with the carbonyl group. In contrast, the desmethyl analogue **1**, whose binding mode has been modeled in analogy to 1ksn, has a suboptimal torsion value just between two minima at about 90° and 180°. This suboptimal torsion angle setting could indeed contribute to the slight loss of affinity. This example highlights how the introduction of a substituent like a methyl group can have an influence on affinity through effects different from a direct molecular interaction.[32] Note that the torsion patterns of **1** and **2** have been defined by means of different terminal atoms. This switch cannot be avoided, since definition via one of the enantiotopic atoms (H in **1** or C in **2**) would lead to an ambiguous assignment of the torsion pattern giving different results depending on the choice of atom that is matched first.

The example can be continued by investigating further modifications of the amide substituent. Conversion of the tertiary carbon atom of **2** into a quaternary system, as exemplified by the change of **3** to **4** in Figure 4, changes the conformation preference from a single, broad minimum at 0° to a system that avoids this syn arrangement. Instead, the three substituents now adopt gauche or anti rotamers.

Torsion Hierarchy Illustrated. The torsion library concept and its realization are best illustrated by an example (Figure 5). The basic substructure of an aryl ether is an aromatic carbon atom connected to a divalent oxygen atom. This pattern only defines the central rotatable bond and is therefore stored in the generic class. The corresponding CSD histogram shows peaks at 0° and 180° and thus indicates a strong preference for the ether substituent to be in plane with the aromatic ring system. A second, less pronounced peak at 90° indicates that orthogonal arrangements are also observed. The generic pattern thus already gives a clear indication of the expected angle ranges. More specific torsion patterns are required to understand if steric or electronic effects of neighboring groups and substituents can strongly shift the conformation preference toward one of the observed angle ranges. Under which circumstances are values of 0°, 90°, or 180° preferred?

The answer is provided in the patterns and histograms in the lower half of Figure 5. Heteroatom substitutions and ortho substituents eliminate one or two of the generic minima. Patterns f–h appear at the bottom of the list; they cover the cases of one, two, or zero ortho substituents that are not further specified. Since the matching of these three patterns is mutually excluding, their positions within the torsion library are exchangeable. Patterns d and e more specifically encode oxygen ortho substituents. They show the same conformational preference as patterns f and g and are thus redundant in the sense that they do not improve the torsion angle annotation of compounds. However, they do add information: A comparison between patterns d and f informs the user that lone pair repulsion of the two neighboring ether oxygen substituents does not change conformation preferences relative to the general case of an undefined substituent. One caveat here is the potentially biased sample space; it might be the case that a

dx.doi.org/10.1021/jm3016816 | J. Med. Chem. 2013, 56, 2016–2028

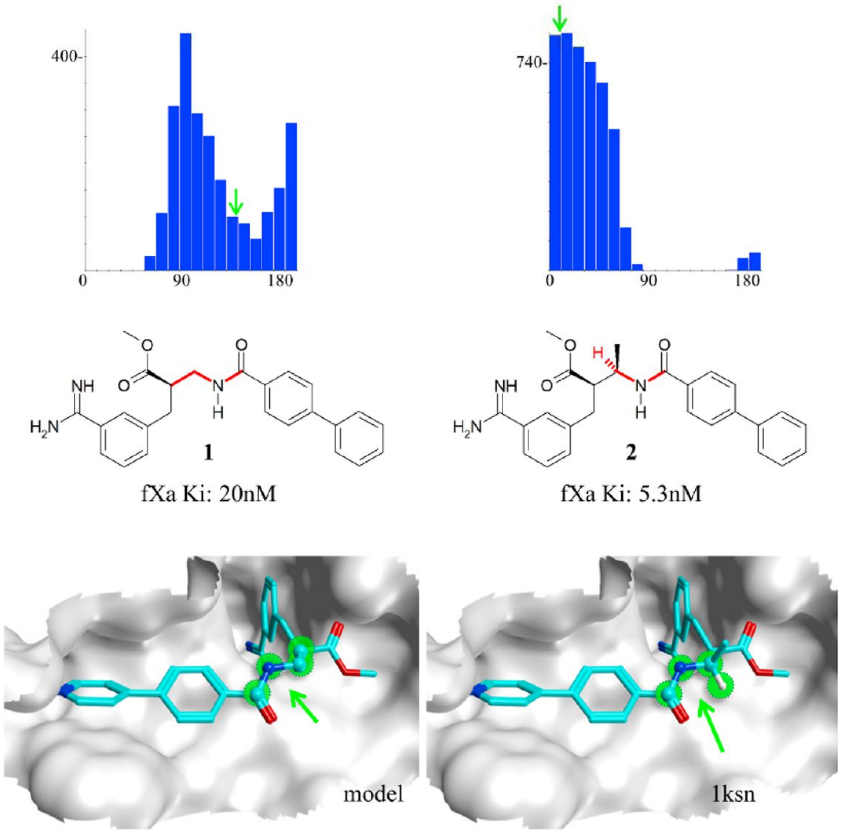


Figure 3. Compounds **1** and **2** are potent factor Xa inhibitors that differ in a single methyl group. Frequency distributions for the torsion angles indicated in red are derived from the CSD. Bottom right: Template for ligand conformation of **2** in PDB structure 1ksn. Bottom left: Modeled binding mode of **1**. The measured torsion angle values of the two ligands are indicated by arrows in the histograms.

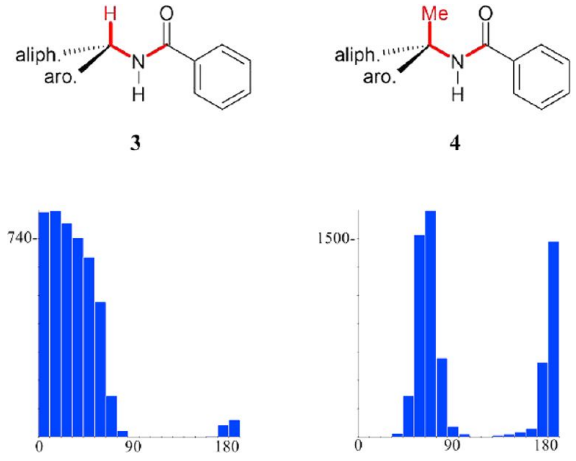


Figure 4. Tertiary substituents of secondary amides have a single, broad minimum equivalent to a syn arrangement between NH and C═O. Quaternary analogues have three accessible rotamers avoiding this syn arrangement.

more generic pattern is dominated by a single more specific one. We have generally opted to retain more specific torsion patterns where a significant number of examples exists, even if the overall predictions are not altered.

In some cases, the location of minima is unaltered but the shape of the histogram is, leading to tighter or broader tolerances. A case in point is pattern (a) in comparison to (e) and (h). Steric bulk at the ether substituent keeps the substituent more strictly in the orthogonal position. Accordingly, the assigned tolerances (tolerance 1/tolerance 2) are different: 20°/30° for pattern (a), 30°/40° for pattern (e), and 30°/35° for pattern (h). For the pyridine derivative pattern c, there is a single peak at 180° indicating that the oxygen and nitrogen lone pairs avoid repulsion.[33] For the pyrimidine analogues pattern b, this repulsion is not avoidable in any rotamer. Will such structures prefer in-plane or out-of-pane conformations? In this case the histogram clearly shows a preference for in-plane rotamers. Note that such questions cannot be generally answered by simple reasoning taking into account steric and electrostatic forces on top of some intrinsic preference, because these factors cannot easily be weighed against each other. For this reason force fields tend to fail for such systems unless they are again specifically parametrized for the case at hand. This illustrates the advantage of the expert system approach.

Figure 6 shows a second example illustrating how changes in the chemical environment can modify torsion preferences in a nonobvious manner. Both substituents at anilinic nitrogen atoms and conjugated double bond systems prefer to be in plane with aromatic systems. There is, however, a significant occupancy over the full range of torsion angle values from 0° to 180°. This is still the case for general anilide and benzamide torsion patterns (pattern d), but the two distributions have a clearly different shape. When only steric effects are separated out as in pattern c, the 90 ± 30° out-of-plane conformations

dx.doi.org/10.1021/jm3016816 | J. Med. Chem. 2013, 56, 2016−2028

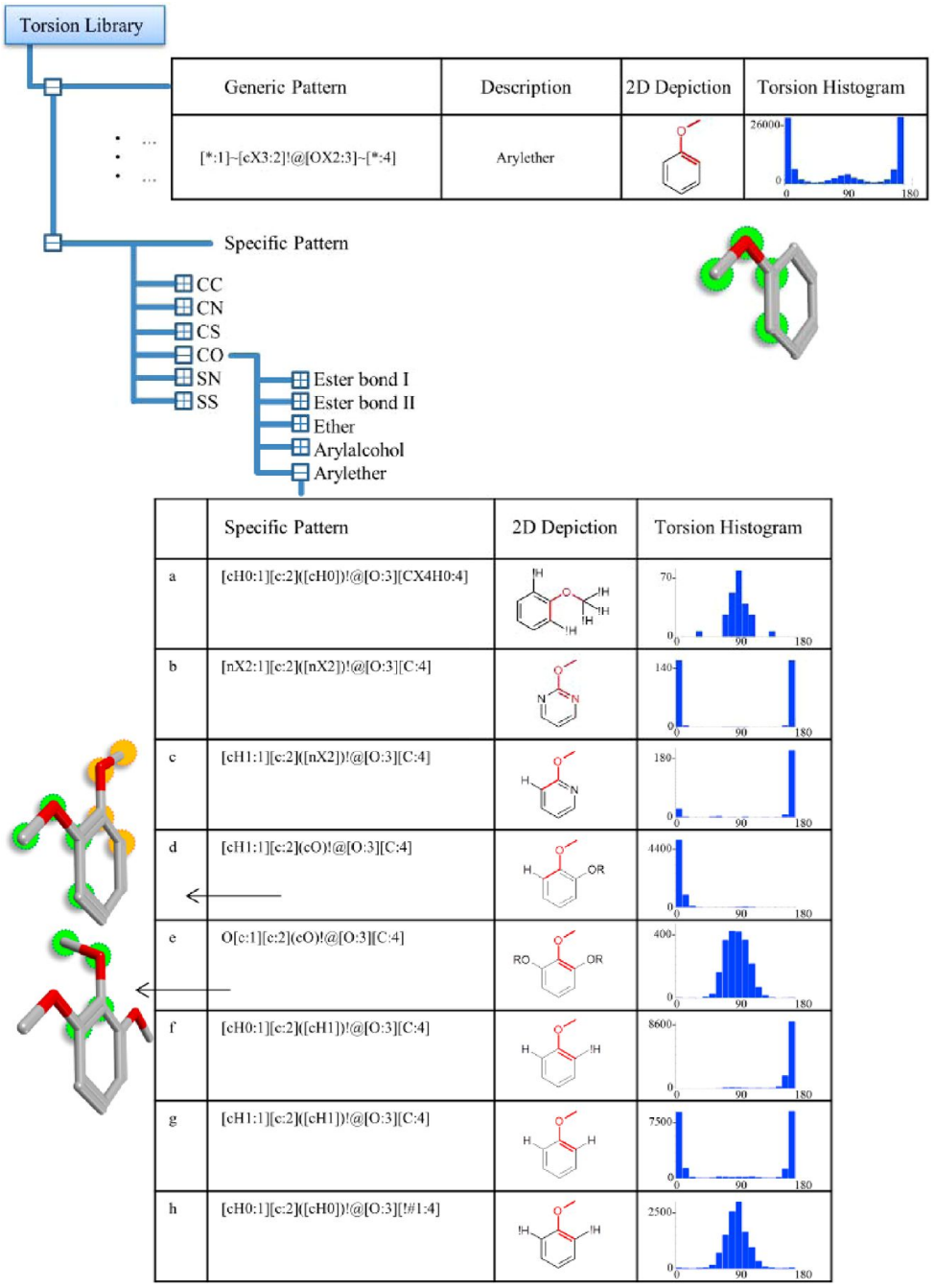


Figure 5. Aryl ethers. The generic SMARTS pattern [cX3:2]!@[OX2:3] shown on top consists of any divalent oxygen atom linked to an aromatic carbon atom. The aryl ether subclass within the top-level class C−O contains specific rules (a−h) for various aromatic ortho substituents as well as ether substituents that have an influence on conformation. Selected structural examples from the CSD are shown. All displayed histograms are derived from CSD data.

disappear. Also, it becomes apparent that anilides and benzamides have slightly different lowest energy rotamers; while anilides tend to be fully in plane, benzamides tend to be slightly out of plane with a twist angle of 30−40°.

The conformational effect of single ortho-F or -Cl substituents on these two systems is quite striking. For the fluorine atom there is a strong preference to be located in plane and on the same side as the amide hydrogen atom. This is true for anilides as well as benzamides (pattern b). CSD-derived torsion preferences agree with calculated torsion profiles (Maestro 9.1, relaxed coordinate scan, HF, 6-31G) superimposed on the histograms (Figure 6); almost all entries are found in the calculated global minima. Interestingly, the fluoroanilides and fluorobenzamides have additional local minima at 120° and 45°, respectively, where one or two CSD entries are found. The chloro-substituted anilides (pattern a) tend to have an in-plane or almost in-plane (0 ± 45°) conformation where the chlorine is preferentially located on the

dx.doi.org/10.1021/jm3016816 | J. Med. Chem. 2013, 56, 2016−2028

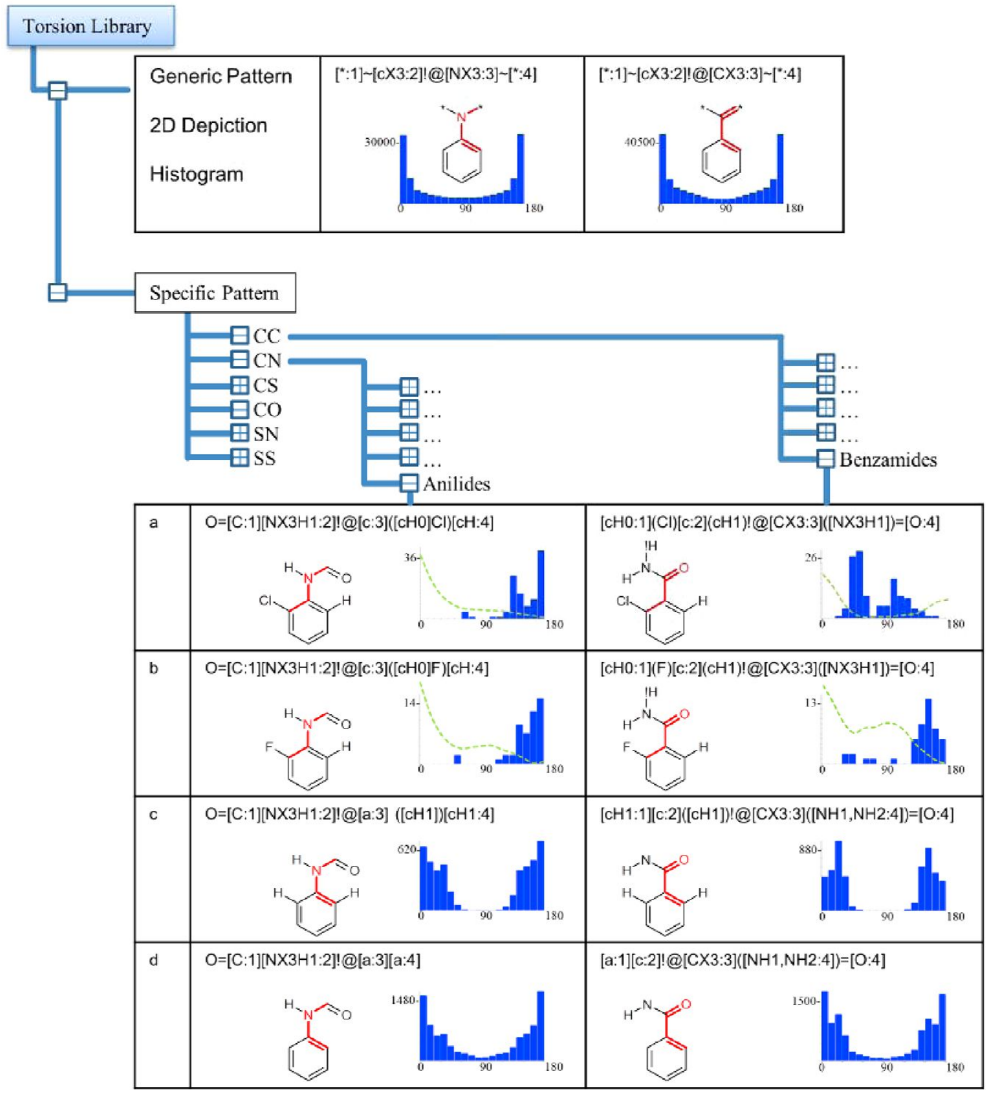


Figure 6. Torsion patterns for nitrogen and sp^2 carbon (both trivalent) bound to aromatic carbon. Both show preferences for in-plane conformations. More specific patterns for the anilides and benzamides provide more detail on trends for more specific substructures. Patterns a and b are overlaid with QM calculated torsion profiles (green dashed lines) scaled to fit the *y* axis.

side of the amide NH group. In the corresponding benzamide case, an in-plane arrangement is no longer observed. Steric strain seems to be the main driver here. The broad occupancy of CSD entries between 30° and 150° nicely correlates with the broad energetic minima in this range of torsion angle values. The calculated torsion profiles in these four cases exemplarily illustrate the reliability of the CSD-derived profiles and show how rules can be further substantiated in case the number of examples is too low for statistically reliable statements.

Coverage of Chemical Space. The current torsion library consists of 97 generic and 393 specific patterns. We wished to understand to what extent these close to 500 rules cover druglike chemical space, whether there are systematic gaps, and in which cases the system would resort to generic rules. For this purpose we used a subset of the ChEMBL library (see Materials and Methods) and counted the number of times each pattern was matched. All rotatable bonds (>5 million) in this subset could be matched to a SMARTS pattern. The majority of 96.3% was matched to a specific pattern; the remainder of 3.7% was matched to a generic pattern. The most frequently used specific patterns describe aliphatic chains, primary amides, aryl ethers, and benzylic substructures (Figure 7). For many of these, sharp peaks are observed, indicating strong conformational preferences. The pattern representing benzylic substructures is somewhat unique, as it shows no conformational preference at all. Some very specific patterns were not matched onto any molecule of the ChEMBL subset, for example, a pattern describing extreme steric bulk for an aryl ether substructure (2-fold ortho substitution in combination with a quaternary, aliphatic carbon ether substituent). The most frequently matched generic pattern (0.6% of the ChEMBL subset rotatable bonds or 757 matches) describes substructures of aliphatic sp^2 carbon atoms connected to aliphatic nitrogen atoms. The corresponding histogram shows sharp peaks at 0° and 180°; thus, a definition of more specific patterns was not deemed necessary. The top 10 generic patterns are included in the Supporting Information. These all show clear conformational preferences that could not be further refined by specific

dx.doi.org/10.1021/jm3016816 | J. Med. Chem. 2013, 56, 2016−2028

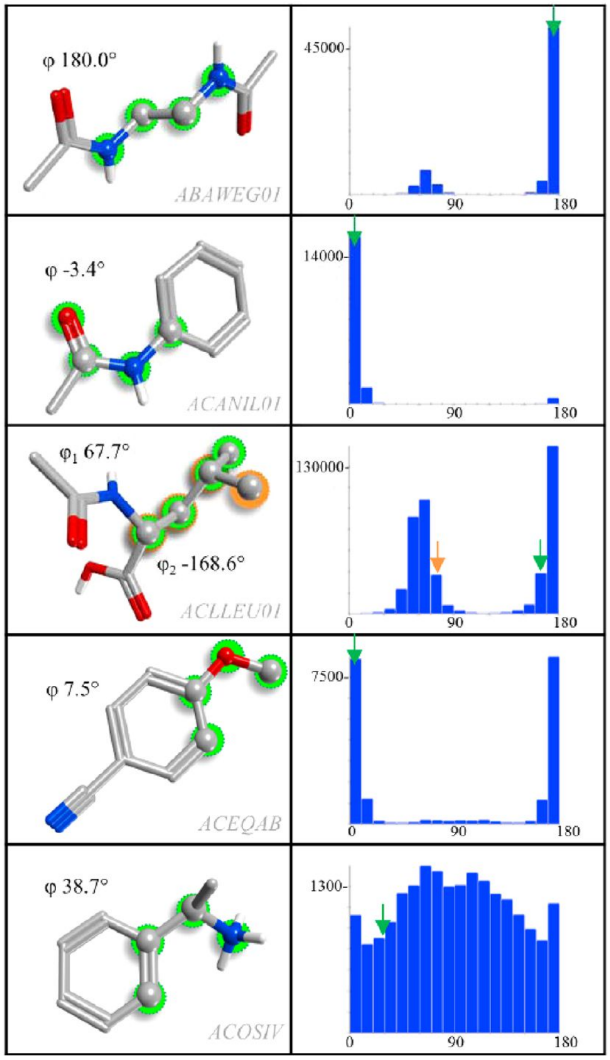


Figure 7. Five specific patterns that are most frequently used to describe aliphatic chains (1 and 3), primary amides (2), aryl ethers (4), and benzyl substructures with nitrogen or oxygen substitution (5). For each pattern an example from the CSD is given with the corresponding torsion angle highlighted and measured. The arrows in the histograms indicate the actual values of measured torsion angles.

definitions, which leads us to conclude that the rule set covers over 98% of the chemical space that is of interest in medicinal chemistry.

Comparing CSD and PDB Histograms. We have not found systematic differences between CSD- and PDB-derived histograms. As has been demonstrated by others, conformational bias due to packing in small molecule crystals is a rare event.[34,35] The occasionally observed differences in conformation preferences are due to sampling bias in the PDB (see the Supporting Information). However, histograms derived from the PDB typically show broader peaks than CSD histograms. This is due to the fact that small molecule conformation preferences often play a subordinate role when macromolecular X-ray structures are fitted to electron densities.[36] Since tolerance values were based on CSD data, this leads to frequent "orange" or "red" torsions when analyzing PDB structures. For the PDB subset approximately 66% of the entries contain at least one rotatable bond that is outside the second tolerance range. In contrast, this is true for only 25% of the CSD entries. Typical candidates are ligands with long, aliphatic carbon chains. Torsion angles along these chains have the well-known minima at ±60° and 180°. The energy difference of 0.9 kcal/mol in favor of the anti conformation is reflected in the peak heights. The corresponding PDB histogram shows a broad shoulder between 60° and 180°. These represent eclipsed conformations that are associated with high energetic penalties that cannot be compensated by typical nonbonded interactions. Analysis of structures like 1cvu[37] (Figure 8 top, resolution 2.4 Å), which displays several eclipsed torsions, shows that a few steps of restricted minimization of the ligand within the rigid binding site can relax these torsions to gauche angles in such a way that the electron density is still satisfied.

Other illustrative cases are ester conformations. Ester alkoxy groups strictly adopt cis rotamers relative to the carbonyl group, as evident from the CSD histogram in Figure 8. The PDB distribution shows the same preference, but the peak extends to angles even beyond 90°. The ligand in the high resolution structure 3rxw[38] may serve as a case in point. Around the ester substituent, no electron density is defined and the ligand seems to have been forced into a conformation that accommodates two small, spherical spots of electron density. The ester substituent is twisted out of plane by 73°. Inspection of the complex structure reveals that this part of the ligand is solvent-exposed and does not interact with the receptor. It is thus doubtful whether the ligand model is a correct interpretation of the electron density.

Like esters, primary amides have a pronounced preference for a syn conformation as shown in the two CSD histograms in Figure 8. It is striking that the PDB histogram for primary amides much better resembles the CSD histogram than the one for esters. This difference might be a direct consequence of using typical force fields for fitting coordinates to electron densities (e.g., the energy difference as calculated with MMFF94 for a 90° out of plane amide is approximately twice as high as that of an ester). PDB entry 3ke1[39] exemplifies that many of the out of plane amide conformations can easily be remodeled to a favorable syn conformation without violating electron density.

We see the refinement of PDB structures as an important application domain for the Torsion Analyzer, since the resulting structural changes are often highly relevant when protein−ligand complex structures are used as a basis for further drug design efforts. We will discuss two examples in more detail here. PDB entry 3tv7[40] has seven rotatable bonds, out of which three have torsion angles outside the second tolerance range (Figure 9): an amide bond in a urea substructure has an almost orthogonal orientation; the neighboring CN bond is in an eclipsed conformation; and the terminal methoxy group is twisted out of plane of the phenyl ring. The measured torsion angle values of the X-ray ligand conformation are located in unpopulated regions in the histograms (red arrows in Figure 9). The authors of the structure mention that the urea NH might interact via a hydrogen bond with the side chain of Asp 216. Closer inspection of the structure shows, however, that all parameters are beyond an acceptable range for a hydrogen bond (distance N···O, 3.4 Å (optimal at 2.8−3.1 Å); angle N−H−O, 129° (optimal at >150°); and the NH group is approximately 80° out of plane with the sp^2 acceptor atom (optimally in plane)). Manual optimization of the ligand

dx.doi.org/10.1021/jm3016816 | J. Med. Chem. 2013, 56, 2016−2028

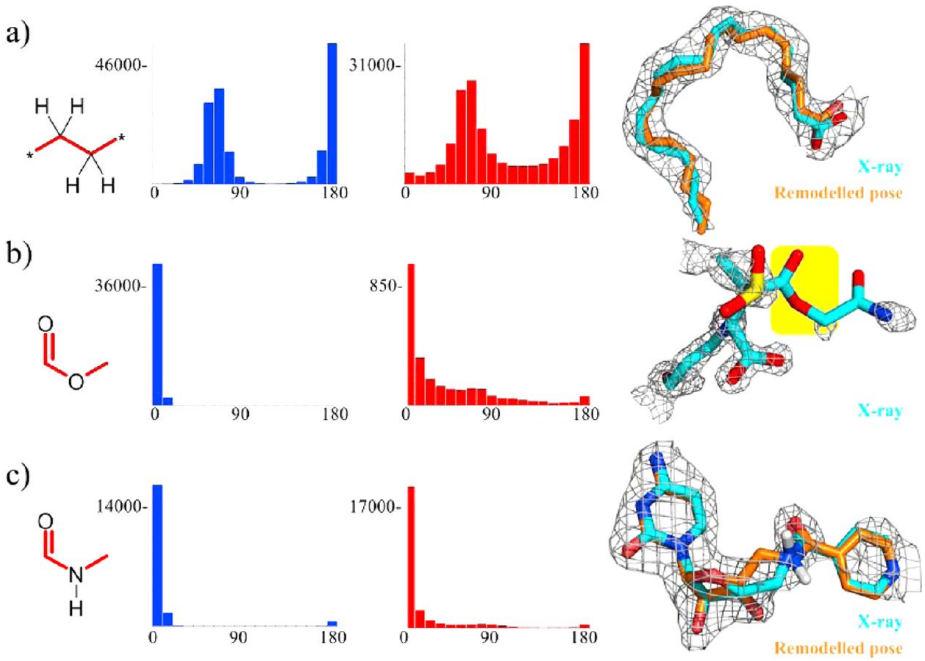


Figure 8. Three selected patterns for the comparison of CSD- and PDB-derived histograms. Conformational preferences from PDB histograms are typically much broader. Each pattern is illustrated by a PDB example. (a) Flexible, aliphatic chains, PDB structure 1cvu (2.4 Å resolution). (b) Esters and PDB structure 3rxw (1.26 Å resolution). There is no significant electron density around the ester substituent. (c) Primary amides, PDB structure 3ke1 (2.4 Å resolution). Twisted conformations can be converted to a fully planar conformation without violating the experimental electron density.

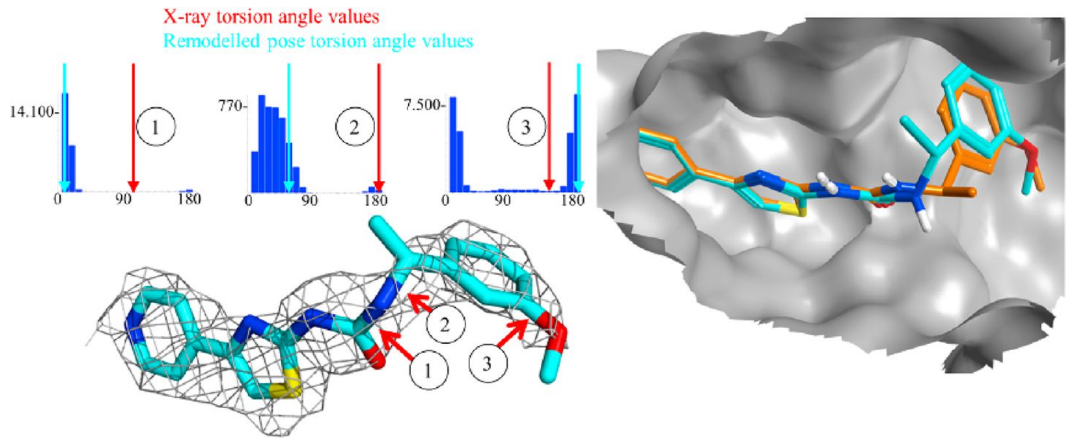


Figure 9. For the bound ligand of PDB entry 3tv7 three bonds are color-coded red (1−3, red arrows). Torsion values of red bonds are marked with red arrows in the corresponding torsion angle value histograms. Rotation around these bonds until they are within the first tolerance interval retains the experimentally determined electron density.

conformation is possible without violating the experimental electron density. The NH still does not interact with binding site residues but is nicely solvent exposed now. The phenyl ring remains in place and thus preserves critical hydrophobic interactions. The only significant difference in shape is the location of the methyl substituent at the tertiary carbon. For this group, no electron density is observed. Altogether we would thus propose a different binding mode for the ligand of 3tv7 that fits well to the experimental electron density, is low in conformational energy, and also offers a favorable orientation for the NH toward solvent-exposed space.

In the second example, the ligand of PDB entry 1gwx[41] has six bonds that are outside the second tolerance range (Figure 10). Manual structural modifications can relocate all of those torsion angles to within the first tolerance range in such a way that the experimental electron density is not violated. The two torsion angles at the tertiary amide (substructure circled in Figure 10) have torsion angle values of approximately 120°, clearly out of range of the second tolerance (90° ± 20°). These two values were manually set to 78° for the cis and 108° for the trans substituent. Even though the modifications in this case are only minor, this could not be accomplished by alternative approaches like, for example, force field minimization, which lead to antiperiplanar conformations for at least one of the nitrogen substituents.

dx.doi.org/10.1021/jm3016816 | J. Med. Chem. 2013, 56, 2016−2028

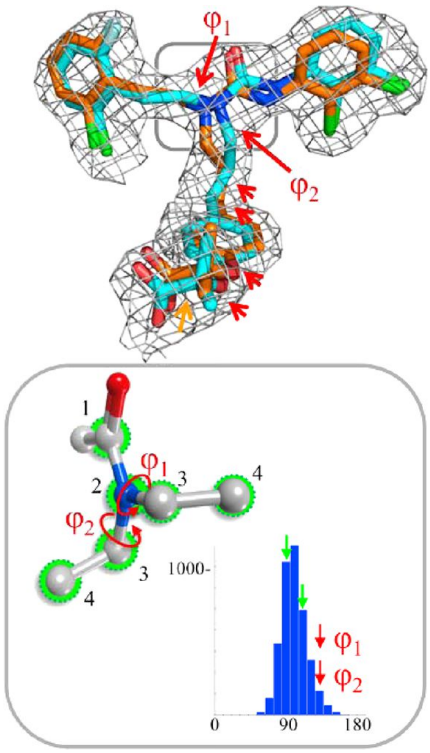


Figure 10. The bound ligand of PDB entry 1gwx (green) is superimposed with a manually refined conformation of the ligand (cyan). Six torsions of the original structure are strained (red arrows). Minor adjustment of these torsions leads to a relaxed structure that still fits the experimentally determined electron density. The terminal carboxylic acid substituent (orange arrow) is left unchanged in order to preserve its interactions in the binding site intact. Bottom: Histogram for the substituents of the type -CH_2R at the secondary amide. There is a strong preference for torsion angles ϕ_1 and ϕ_2 to be around ±90°.

MATERIALS AND METHODS

Here we describe technical aspects of the concept of torsion patterns. We first explain the notation used to encode torsion patterns and how these patterns are matched to molecular structures. Then we outline the hierarchy concept of the torsion library, the generation of torsion histograms from PDB and CSD subsets, and the derivation of torsion rules from these histograms. We then list key technical aspects of the Torsion Analyzer tool that manages all these individual steps.

Torsion Patterns. Torsion patterns are described by SMARTS line notation.[42] The central two atoms are linked by a rotatable bond, which here is defined as any acyclic bond (defined by "!@"; the restriction to a single bond is done in the matching process). In addition to the four-atom sequence defining the torsion angle, additional substructure elements may be encoded in the SMARTS pattern. To increase flexibility, we made use of a SMARTS extension that describes the hybridization state sp^3, sp^2, or sp by the expressions ^3, ^2, and ^1, respectively.[43,44] In addition, the primitive N_lp is introduced to define torsion angles involving the lone pair of nitrogen atoms. This allows us to unambiguously define torsions involving sp^3-hybridized, trivalent nitrogen atoms at rotatable bonds like they are, for example, present in sulfonamides. Such systems would otherwise require two separate matching steps of N substituent atoms. We emphasize that the primitive N_lp is only valid for sp^3-hybridized nitrogen atoms.

The assignment of a torsion pattern to a specific torsion angle is done by means of a SMARTS matching algorithm. Out of the many approaches that have been published,[45−49] we use a modified version of the VF2 algorithm,[49] since it is reported to be the fastest on molecular data.[50]

Torsion Library. All torsion patterns are stored in a torsion library and ordered in a hierarchical fashion. At the highest level, this hierarchy classifies torsion patterns according to the elements of their central bonds (atoms 2 and 3 in Figure 1). Only the elements C, N, O, and S are taken into consideration for specific classes. Out of the 10 possible combinations, six were deemed to be sufficient to describe rotatable bonds in druglike chemical space forming their own specific class; these are the carbon bonds CC, CN, CO, and CS as well as the heteroatom bonds NS and SS. A seventh class covers all generic torsion patterns of the previous six classes as well as generic descriptions of other heteroatom combinations and is abbreviated with GG (each G represents one generically described atom at the center of the rotatable bond; atoms 1 and 4 are defined by the SMARTS primitive [*]). Class GG includes all possible pair combinations between the elements C, N, O, S in their aliphatic and aromatic versions as well as possible valence states, for example, between aromatic carbon and aliphatic carbon with three and four bonds, respectively ([cX3:2]!@[CX4:3]). It further contains the most generic patterns described by hybridization states only, for example, between any sp^3 and sp^2 hybridized atoms ([*^3:2]!@[*^2:3]) and, in addition, a single subclass for rotatable bonds between two aromatic ring systems [a:2]!@[a:3]. Class GG alone thus covers the entire chemical space in a generic fashion; its rules are only applied when no matching pattern exists in one of the specific classes.

Each of the six specific classes contains several subclasses covering typical functional groups or frequently occurring substructures. Currently the seven main classes contain 35 subclasses in total. The class CO, for example, is further subdivided into an ester, ether, and aryl ether subclass. Each subclass in turn can contain further subclasses. In Figures 5 and 6 we have illustrated the hierarchy within the torsion library system. Within each class or subclass, the torsion data are ordered by decreasing specificity, with the most specific patterns placed at the top. The hierarchical system has been structured and labeled such that it can be easily understood and modified. At the same time, it minimizes the time requirements for the actual SMARTS matching process. To add visual support when creating or editing SMARTS patterns, we incorporated the SMARTSviewer[51] into the Torsion Analyzer.

The matching process for a given molecule proceeds as follows: (1) Atoms 2 and 3 of the torsion angle are used to identify the main class. If none is found, the generic class GG is used. (2) The torsion patterns in the class are probed from top to bottom. The search stops at the first matching pattern. If no pattern is found, the search algorithm looks for a matching subclass. If there is no matching subclass, the search continues in the GG class. (3) If a matching subclass is found, the search iteratively proceeds as in the previous step until a matching pattern is identified; otherwise, the generic class GG is used. (4) Probing in the generic class GG starts with probing against subclass [a][a]. If this subclass does not match, torsion patterns are probed from top to bottom until the first matching pattern is found. The hybridization-only patterns ([*:1] [*^3:2]!@[*^3:3] [*:4], [*:1] [*^3:2]!@[*^2:3] [*:4], [*:1] [*^2:2]!@[*^2:3] [*:4]) at the bottom of this class ensure that every possible torsion angle can at least be matched to one torsion pattern. If there is more than one way to match a pattern onto a specific bond, for example, if atom 1 and/or atom 4 in the pattern is defined as any atom ([*]), we need a process to ensure a unique match with respect to the atoms 1−4. Therefore, all possible matches are examined, corresponding torsion angles are calculated, and the match resulting in a torsion angle nearest a histogram peak is used as the unique match.

Generation and Analysis of Torsion Histograms. Each torsion pattern in the library is associated with histograms derived from small molecule crystal information contained in the CSD and the PDB databases. The subsets of the CSD[19] and the extraction of ligands from the PDB[20] used for this purpose will be described further below.

For each torsion pattern and for both CSD and PDB, frequency distributions were generated in the form of histograms. Torsion values were recorded from −180° to +180° in 10° bins, considering all

dx.doi.org/10.1021/jm3016816 | J. Med. Chem. 2013, 56, 2016−2028

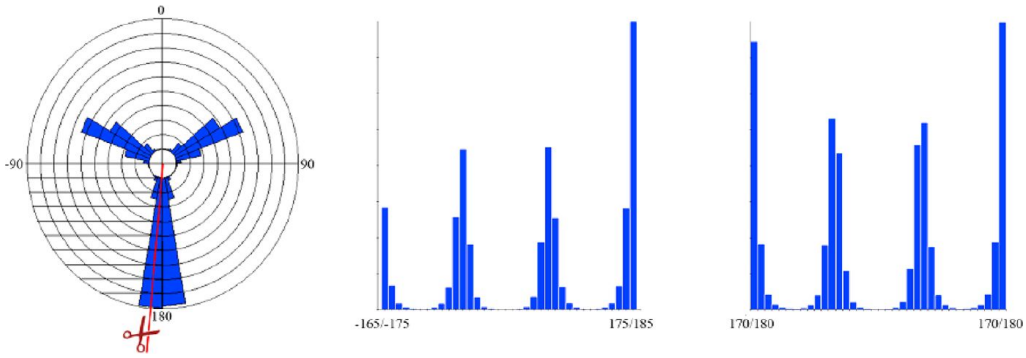


Figure 11. Left: Polar histograms for torsion angles can be interpreted like Newman projections. The red line indicates the cut for the linear representation. Center: Histogram as used for automated analysis. Note that this shift introduces asymmetry of the linear representation. Right: Histogram as displayed in the Torsion Analyzer.

possible matches of a pattern for each molecule in the data set. In a condensed view we also show folded histograms where only absolute values are shown. Most of the torsion patterns are associated with minima centered at 0° or multiples of 10°. We have defined bins such that they are centered at these values, as this leads to sharper peaks that can be readily analyzed. The central bin ranges from −5° to +5°. As a consequence, terminal bins range from 175° to 185° and from −175° to −185°, respectively. Since angles above an absolute value of 180° do not occur, this effectively halves the size of the terminal bins. To account for this, we record angles from −180° to −175° in the 175−185° bin. Effectively, this process makes use of the cyclic nature of the angle histograms. It is equivalent to cutting a cyclic histogram between the two bins [−165°, −175°] and [−175°, 175°] (Figure 11).

As a consequence, histograms with peaks at 180° appear unsymmetrical in a linear representation, and for display purposes, we use histograms shifted by 5° or in a "folded" form for which negative angle values were set to the corresponding positive ones. This folding step is valid under the assumption that chirality does not play a role, as is the case for all patterns of the current torsion library.

Histogram peaks, representing the most frequently observed torsion values, were automatically extracted from histograms with more than 100 data points and defined as the local minima of the torsion distributions. The total count of the histogram bins was converted to a relative count (percent values). The central angles of all bins containing more than 4% of the histogram data were defined as minima for the respective torsion. Around these minima, two tolerance intervals were defined, representative of the width of the peak. Tolerances were assigned automatically for histograms with more than 100 data points. The inner tolerance interval was defined by identifying the first neighboring histogram peaks to the left and to the right of the minimum that contained less than 2.5% of the data and calculating the distance of this bin's central angle to the minimum angle. Where these values differed, the smaller range was chosen. The second tolerance interval was calculated in the same way but with a lower cutoff of 1.5%. If the histogram data were evenly distributed over all bins (variance of <0.1), a 30° grid from −180° to +180° was used as a default set of minima, with 10° and 15° as the first and second tolerance interval. The method we used to extract the most frequently observed torsion values is similar to the one Sadowski and Boström used to automatically generate torsion rules from crystal structures.[24] These authors used histograms with 30° bins, a minimum of 20 hits per histogram, and a minimum of 45% frequency for each bin.

Where no data points were available for a histogram, the torsion pattern was kept in the library but set to inactive. For histograms with more than 0 but less than 100 data points, torsion minima and tolerance values were defined manually. If distinct peaks could be identified, they were used to define the local minima of the torsion distribution. For these minima, broader tolerance intervals between 20° and 30° were defined. If the data points were too scattered, either a default 30° grid (10° and 15° as the first and second tolerance interval) or the torsion pattern was set to inactive. About 15% of the patterns in the current torsion library have histograms with less than 100 data points.

Implementation Notes. We developed a new software library and a graphical user interface to analyze small molecule conformations. The software library is implemented in C and C++ and includes the necessary components for handling a torsion library, generating and analyzing torsion histograms, and analyzing conformations. Reading, writing, and handling of molecule data are done with the NAOMI software library.[52] The torsion library including all defined and extracted data (SMARTS pattern, torsion histograms, torsion minima, and tolerances) is stored in an XML file. A specifically defined XML scheme is used to validate the torsion library file. Reading and writing of XML data are done using the libxml2 software library. The graphical user interface (GUI) is implemented in C and C++ using the portable Qt GUI toolkit. The choice of implementation language and GUI toolkit ensures that the software can be compiled and run on a variety of platforms, e.g., Linux or Microsoft Windows.

CSD Data Set. The CSD subset was generated by extracting all entries with an associated 3D structure and at least one carbon atom from the 2011 version of the CSD database (approximately 580 000 entries, ConQuest, version 1.14). Furthermore, entries with elements other than H, C, N, O, F, Cl, Br, I, S, and P were removed. Ions, powder structures, organometallic compounds, and structures with an *R*-factor of less than 10% were omitted from the search. This resulted in approximately 145 000 structures that were exported in mol2 format. These structures were further processed by Corina (version 3.46) with the driver options "no3d, newtypes, rs" to assign consistent atom types. The output format was set to mol2. The final CSD subset contains approximately 140 000 structures and represents 24% of all CSD entries.

PDB Data Set. For the PDB histograms, we extracted conformations of protein-bound ligands from Proasis2,[53] a curated version of the PDB. The ligand structures were taken from all HET entries, except metals and commonly found small ions, in the PDB as of December 8, 2008. Ligands with less than 5 or more than 100 atoms were removed, excluding in particular large peptidic groups. This resulted in a database of 77 065 ligands derived from 24 163 PDB entries.

ChEMBL Data Set. A subset of ChEMBL (version 13) was derived by querying for compounds with a molecular weight between 200 and 500 of the approximately 920 000 compounds. The subset was converted with Corina (version 3.46) and flags "rs,wh" to 3D sdf format.

SUMMARY AND OUTLOOK

We have presented an exhaustive collection of torsion patterns associated with conformation rules in combination with a user interface designed for efficient access to this information. The traffic-light coloring scheme gives an immediate impression of the likelihood of a given conformation to occur. For each

dx.doi.org/10.1021/jm3016816 | J. Med. Chem. 2013, 56, 2016−2028

combination of torsion pattern and rule set, there is a direct link to the underlying torsion angle distributions from the CSD and PDB, allowing the user to understand the basis and reliability of the rules, to judge an individual conformation within the context of the entire rotameric profile, and to highlight differences in data quality between the two data sources.

The expert system approach is a way of eliminating difficulties and error sources that may arise from ad hoc rule extraction from the CSD through its interfaces ConQuest[22] or its automated sister tool Mogul.[21] Database searches with ConQuest need to be set up such that substructures are specific enough to tease out relevant conformation preferences but on the other hand general enough to achieve significance of the results. Mogul, in contrast, defines a chemical environment in an automated fashion that cannot be modified by the user. Our approach pre-encodes the expert's approach to such searches while maintaining full transparency about the relationship between extracted knowledge and the underlying data. This approach has the further advantage that rules can be added from orthogonal sources of information, for example, from quantum chemical calculations. Chemical space accessed and utilized by medicinal chemists has been growing over the years and has not been exhaustively explored yet. New chemical substructures tend to appear in crystal structure databases with some delay. An example is the use of the trifluoromethoxy group. The first crystal structure with this substructure was added to the CSD in 1987. Over the next 20 years approximately one trifluoromethoxy-substituted molecule was added to the CSD per year. Then submissions increased to about eight per year such that today the 90° out of plane conformation preference is clearly visible. In such cases, calculated torsion profiles can bridge the gap until statistical significance is reached.

The current torsion library has a number of limitations that could be addressed by extensions of the way patterns are defined and employed. First, our analysis is currently based on the assumption of independent torsion angles. There are, however, clear cases where two consecutive torsion angles need to be looked at simultaneously for clear trends to emerge. The most prominent examples of this type are aryl-X-aryl systems;[25] an analogous one is the tertiary amides discussed above. This extension will become part of a future release of the Torsion Analyzer. Similarly, a full conformation analysis is not complete without an analysis of ring systems. Rings of course contain dependent torsion angles as well, and methods have been published to classify the wealth of crystal structure information for flexible rings.[54] We have previously introduced the concept of torsion fingerprints,[55] which could be extended in this regard.

All current torsion patterns have been defined as achiral substructures. In many cases, chirality will not dramatically shift torsion preferences, and in many cases, conformations of chiral molecules can be analyzed by means of achiral patterns, as has been done in the examples of Figures 3 and 4 above. In the general case, neglecting chirality is of course a simplification of reality. Interestingly, chirality plays a role in the definition of torsion patterns even when the substructure at hand is not chiral yet. Figure 12 shows this for patterns of increasing complexity. While there is only one way of defining a torsion angle of an aryl ether (a), unambiguous assignment of torsion profiles to a sulfonamide (b) requires the choice of nitrogen as the terminal atom. Picking one of the two prochiral oxygen atoms as the terminal atom would lead to an introduction of chirality into the pattern, and thus, the locations of the nitrogen and the second oxygen atom to each other are not clearly defined. For the substructure in Figure 12c a chiral SMARTS pattern would be required to derive a torsion profile. This profile cannot be folded, since it would be asymmetric with respect to the origin. Finally, there is a notable difference between Figure 12c and Figure 12d. The symmetry of the phenyl ring in Figure 12d leads to the additional complication that two alternative torsion angle matches are possible. Because of the chirality of the substituent, these are no longer equivalent, so a single torsion pattern is not sufficient anymore. We have avoided such complications by always choosing the smallest substituent of a chiral pattern as the terminal atom. This leads to an optimum overlap of the two mirror image histograms such that steric effects can be quite well accounted for. Future versions of the Torsion Analyzer might deal with chiral patterns in a more differentiated fashion.

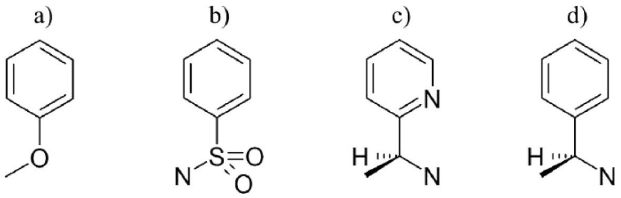


Figure 12. Chirality plays a role in the definition of torsion patterns even if the substructure is not yet chiral. Shown are selected achiral, prochiral, and chiral substructures.

■ ASSOCIATED CONTENT

Supporting Information

Histograms for the 10 generic patterns that are most frequently used plus a corresponding molecule from the CHEMBL data set (S1); example of sampling bias (S2); and a separate file containing the full torsion library in txt format. This material is available free of charge via the Internet at http://pubs.acs.org.

■ AUTHOR INFORMATION

Corresponding Author

*For T.S.-G.: phone, 0041-61-6888309; e-mail, tanja.schulz-gasch@roche.com. For M.R.: phone, 0049-40-428387351; e-mail, rarey@zbh.uni-hamburg.de.

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

We gratefully acknowledge the active support and feedback of many colleagues in medicinal chemistry and CADD. We thank Bernd Kuhn for the QM profiles from Figure 6 and Jérôme Hert for support with the ChEMBL subset. We are grateful to Prof. Klaus Müller for pioneering the systematic growth of empirical knowledge in structure-based design. We also thank BiosolveIT GmbH for help with GUI development and software components. C.S. thanks F. Hoffmann-La Roche Ltd. for financial support.

■ ABBREVIATIONS USED

CSD, Cambridge Structural Database; QM, quantum mechanics; GUI, graphical user interface; 3D, three-dimensional; MMFF94, Merck molecular force field 94; XML, extensible markup language

■ ADDITIONAL NOTE

A prototype of the Torsion Analyzer is available at http://www.biosolveit.de/TorsionAnalyzer.

■ REFERENCES

(1) Hawkins, P. C. D.; Skillman, A. G.; Warren, G. L.; Ellingson, B. A.; Stahl, M. T. Conformer Generation with OMEGA: Algorithm and Validation Using High Quality Structures from the Protein Databank and Cambridge Structural Database. *J. Chem. Inf. Model.* **2010**, *50*, 572−584.

(2) Watts, K. S.; Dalal, P.; Murphy, R. B.; Sherman, W.; Friesner, R. A.; Shelley, J. C. ConfGen: A Conformational Search Method for Efficient Generation of Bioactive Conformers. *J.Chem. Inf. Model.* **2010**, *50*, 534−546.

(3) *MacroModel*, version 9.6; Schrödinger, LLC: New York, NY, 2012; http://www.schrodinger.com/productpage/14/11/ (accessed Oct 18, 2012).

(4) Mohamadi, F.; Richards, N. G. J.; Guida, W. C.; Liskamp, R.; Lipton, M.; Caufield, C.; Chang, G.; Hendrickson, T.; Still, W. C. MacroModel—An Integrated Software System for Modeling Organic and Bioorganic Molecules Using Molecular Mechanics. *J. Comput. Chem.* **1990**, *11*, 440−467.

(5) Chang, G.; Guida, W. C.; Still, W. C. An Internal-Coordinate Monte Carlo Method for Searching Conformational Space. *J. Am. Chem. Soc.* **1989**, *111*, 4379−4386.

(6) Ferguson, D. M.; Raber, D. J. A New Approach to Probing Conformational Space with Molecular Mechanics: Random Incremental Pulse Search. *J. Am. Chem. Soc.* **1989**, *111*, 4371−4378.

(7) Crippen, G. M. Note Rapid Calculation of Coordinates from Distance Matrices. *J. Comput. Phys.* **1978**, *26*, 449−452.

(8) Izrailev, S.; Zhu, F.; Agrafiotis, D. K. A Distance Geometry Heuristic for Expanding the Range of Geometries Sampled during Conformational Search. *J. Comput. Chem.* **2006**, *27*, 1962−1969.

(9) Rusinko, A., III; Sheridan, R. P.; Nilakantan, R. Using CONCORD To Construct a Large Database of Three-Dimensional Coordinates from Connection Tables. *J. Chem. Inf. Comput. Sci.* **1989**, *29*, 251−255.

(10) Gasteiger, J.; Rudolph, C.; Sadowski, J. Automatic Generation of 3D-Atomic Coordinates for Organic Molecules. *Tetrahedron Comput. Methodol.* **1990**, *3*, 537−547.

(11) Kolossváry, I.; Guida, W. C. Low Mode Search. An Efficient, Automated Computational Method for Conformational Analysis: Application to Cyclic and Acyclic Alkanes and Cyclic Peptides. *J. Am. Chem. Soc.* **1996**, *118*, 5011−5019.

(12) Kolossváry, I.; Guida, W. C. Low-Mode Conformational Search Elucidated: Application to C39H80 and Flexible Docking of 9-Deazaguanine Inhibitors into PNP. *J. Comput. Chem.* **1999**, *20*, 1671−1684.

(13) Rarey, M.; Kramer, B.; Lengauer, T. Time-Efficient Docking of Flexible Ligands into Active Sites of Proteins. *Proc. Int. Conf. Intell. Syst. Mol. Biol.* **1995**, *3*, 300−308.

(14) Friesner, R. A.; Banks, J. L.; Murphy, R. B.; Halgren, T. A.; Klicic, J. J.; Mainz, D. T.; Repasky, M. P.; Knoll, E. H.; Shelley, M.; Perry, J. K.; Shaw, D. E.; Francis, P.; Shenkin, P. S. Glide: A New Approach for Rapid, Accurate Docking and Scoring. 1. Method and Assessment of Docking Accuracy. *J. Med. Chem.* **2004**, *47*, 1739−1749.

(15) Halgren, T. A.; Murphy, R. B.; Friesner, R. A.; Beard, H. S.; Frye, L. L.; Pollard, W. T.; Banks, J. L. Glide: A New Approach for Rapid, Accurate Docking and Scoring. 2. Enrichment Factors in Database Screening. *J. Med. Chem.* **2004**, *47*, 1750−1759.

(16) Dixon, S. L.; Smondyrev, A. M.; Knoll, E. H.; Rao, S. N. PHASE: A New Engine for Pharmacophore Perception, 3D QSAR Model Development, and 3D Database Screening: 1. Methodology and Preliminary Results. *J. Comput.-Aided Mol. Des.* **2006**, *20*, 647−671.

(17) Rush, T. S.; Grant, J. A.; Mosyak, L.; Nicholls, A. A Shape-Based 3-D Scaffold Hopping Method and Its Application to a Bacterial Protein−Protein Interaction. *J. Med. Chem.* **2005**, *48*, 1489−1495.

(18) Grant, J. A.; Gallardo, M. A.; Pickup, B. T. A fast Method of Molecular Shape Comparison: A Simple Application of a Gaussian Description of Molecular Shape. *J. Comput. Chem.* **1998**, *17*, 1653−1666.

(19) Allen, F. H. The Cambridge Structural Database: A Quarter of a Million Crystal Structures and Rising. *Acta Crystallogr.* **2002**, *B58*, 380−388.

(20) Berman, H. M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T. N.; Weissig, H.; Shindyalov, I. N.; Bourne, P. E. The Protein Data Bank. *Nucleic Acids Res.* **2000**, *28*, 235−242.

(21) *Mogul—A Knowledge Base of Molecular Geometry*, version 3.0; The Cambridge Crystallographic Data Centre: Cambridge, U.K.; http://www.ccdc.cam.ac.uk/products/csd_system/mogul/ (accessed Oct 15, 2012).

(22) *ConQuest—The Interface for the CSD System*, version 1.14; The Cambridge Crystallographic Data Centre: Cambridge, U.K.; http://www.ccdc.cam.ac.uk/products/csd_system/conquest/ (accessed Oct 15, 2012).

(23) Klebe, G.; Mietzner, T. A Fast and Efficient Method To Generate Biologically Relevant Conformations. *J. Comput.-Aided Mol. Des.* **1994**, *8*, 583−606.

(24) Sadowski, J.; Boström, J. MIMUMBA Revisited: Torsion Angle Rules for Conformer Generation Derived from X-ray Structures. *J. Chem. Inf. Model.* **2006**, *46*, 2305−2309.

(25) Brameld, K. A.; Kuhn, B.; Reuter, D. C.; Stahl, M. Small Molecule Conformational Preferences Derived from Crystal Structure Data. A Medicinal Chemistry Focused Analysis. *J. Chem. Inf. Model.* **2008**, *48*, 1−24.

(26) Bissantz, C.; Kuhn, B.; Stahl, M. A Medicinal Chemist's Guide to Molecular Interactions. *J. Med. Chem.* **2010**, *53*, 5061−5084.

(27) Hardegger, L. A.; Kuhn, B.; Spinnler, B.; Anselm, L.; Ecabert, R.; Stihle, M.; Gsell, B.; Thoma, R.; Diez, J.; Benz, J.; Plancher, J.-M.; Hartmann, G.; Banner, D. W.; Haap, W.; Diederich, F. Systematic Investigation of Halogen Bonding in Protein−Ligand Interactions. *Angew. Chem., Int. Ed.* **2011**, *50*, 314−318.

(28) Kuhn, B.; Fuchs, J. E.; Reutlinger, M.; Stahl, M.; Taylor, N. R. Rationalizing Tight Ligand Binding through Cooperative Interaction Networks. *J. Chem. Inf. Model.* **2011**, *51*, 3180−3198.

(29) Kuhn, B.; Mohr, P.; Stahl, M. Intramolecular Hydrogen Bonding in Medicinal Chemistry. *J. Med. Chem.* **2010**, *53*, 2601−2611.

(30) Czekaj, M.; Klein, S. I.; Guertin, K. R.; Gardner, C. J.; Zulli, A. L.; Pauls, H. W.; Spada, A. P.; Cheney, D. L.; Brown, K. D.; Colussi, D. J.; Chu, V.; Leadley, R. J.; Dunwiddie, C. T. Optimization of the beta-Aminoester Class of Factor Xa Inhibitors. Part 1: P4 and Side-Chain Modifications for Improved in Vitro Potency. *Bioorg. Med. Chem. Lett.* **2002**, *12*, 1667−1670.

(31) Guertin, K. R.; Gardner, C. J.; Klein, S. I.; Zulli, A. L.; Czekaj, M.; Gong, Y.; Spada, A. P.; Cheney, D. L.; Maignan, S.; Guilloteau, J.-P.; Brown, K. D.; Colussi, D. J.; Chu, V.; Heran, C. L.; Morgan, S. R.; Bentley, R. G.; Dunwiddie, C. T.; Leadley, R. J.; Pauls, H. W. Optimization of the beta-Aminoester Class of Factor Xa Inhibitors. Part 2: Identification of FXV673 as a Potent and Selective Inhibitor with Excellent in Vivo Anticoagulant Activity. *Bioorg. Med. Chem. Lett.* **2002**, *12*, 1671−1674.

(32) Leung, C. S.; Leung, S. S. F.; Tirado-Rives, J.; Jorgensen, W. L. Methyl Effects on Protein−Ligand Binding. *J. Med. Chem.* **2012**, *55*, 4489−4500.

(33) Chein, R.-J.; Corey, E. J. Strong Conformational Preferences of Heteroaromatic Ethers and Electron Pair Repulsion. *Org. Lett.* **2010**, *12*, 132−135.

(34) Cruz-Cabeza, A. J.; Liebeschuetz, J. W.; Allen, F. H. Systematic Conformational Bias in Small-Molecule Crystal Structures Is Rare and Explicable. *CrystEngComm* **2012**, *14*, 6797−6811.

(35) Pascal, R. A.; Wang, C. M.; Wang, G. C.; Koplitz, L. V. Ideal Molecular Conformation versus Crystal Site Symmetry. *Cryst. Growth Des.* **2012**, *12*, 4367−4376.

(36) Warren, G. L.; Do, T. D.; Kelley, B. P.; Nicholls, A.; Warren, S. D. Essential Considerations for Using Protein−Ligand Structures in Drug Discovery. *Drug Discovery Today* **2012**, *17*, 1270−1281.

(37) Kiefer, J. R.; Pawlitz, J. L.; Moreland, K. T.; Stegeman, R. A.; Hood, W. F.; Gierse, J. K.; Stevens, A. M.; Goodwin, D. C.; Rowlinson, S. W.; Marnett, L. J.; Stallings, W. C.; Kurumbail, R. G. Structural Insights into the Stereochemistry of the Cyclooxygenase Reaction. *Nature* **2000**, *405*, 97−101.
(38) Ke, W.; Bethel, C. R.; Papp-Wallace, K. M.; Pagadala, S. R. R.; Nottingham, M.; Fernandez, D.; Buynak, J. D.; Bonomo, R. A.; van den Akker, F. Crystal Structures of KPC-2 beta-Lactamase in Complex with 3-Nitrophenyl Boronic Acid and the Penam Sulfone PSR-3-226. *Antimicrob. Agents Chemother.* **2012**, *56*, 2713−2718.
(39) Edwards, T. E.; Davies, D. R.; Hartley, R.; Zeller, W. Crystal Structure of 2C-Methyl-D-erythritol 2,4-Cyclodiphosphate Synthase from *Burkholderia pseudomallei* in Complex with a Fragment−Nucleoside Fusion D000161829. http://www.rcsb.org/pdb/explore/explore.do?structureId=3ke1 (accessed Dec 4, 2012).
(40) Pireddu, R.; Forinash, K. D.; Sun, N. N.; Martin, M. P.; Sung, S.-S.; Alexander, B.; Zhu, J.-Y.; Guida, W. C.; Schonbrunn, E.; Sebti, S. M.; Lawrence, N. J. Pyridylthiazole-Based Ureas as Inhibitors of Rho Associated Protein Kinases (ROCK1 and 2). *Med. Chem. Commun.* **2012**, *3*, 699−709.
(41) Xu, H. E.; Lambert, M. H.; Montana, V. G.; Parks, D. J.; Blanchard, S. G.; Brown, P. J.; Sternbach, D. D.; Lehmann, J. M.; Wisely, G. B.; Willson, T. M.; Kliewer, S. A.; Milburn, M. V. Molecular Recognition of Fatty Acids by Peroxisome Proliferator-Activated Receptors. *Mol. Cell* **1999**, *3*, 397−403.
(42) *Daylight Theory Manual: SMARTS—A Language for Describing Molecular Patterns*; Daylight Chemical Information Systems: Laguna Niguel, CA; http://www.daylight.com/dayhtml/doc/theory/theory.smarts.html (accessed Nov 1, 2012).
(43) *SMARTS Pattern Matching*; OpenEye Scientific Software: Cambridge, MA; http://www.eyesopen.com/docs/toolkits/current/html/OEChem_TK-python/SMARTS.html (accessed Aug 06, 2012).
(44) *SMARTS*; Open Babel; http://openbabel.org/wiki/SMARTS (accessed Aug 06, 2012).
(45) Ray, L. C.; Kirsch, R. A. Finding Chemical Records by Digital Computers. *Science* **1957**, *126*, 814−819.
(46) Sussenguth, E. H. A Graph-Theoretic Algorithm for Matching Chemical Structures. *J. Chem. Doc.* **1965**, *5*, 36−43.
(47) Ullmann, J. R. An Algorithm for Subgraph Isomorphism. *J. Assoc. Comput. Mach.* **1976**, *23*, 31−42.
(48) Cordella, L.; Foggia, P.; Sansone, C.; Vento, M. Performance Evaluation of the VF Graph Matching Algorithm. *Image Anal. Process.* **1999**, 1172−1177.
(49) Cordella, L. P.; Foggia, P.; Sansone, C.; Vento, M. A (Sub)Graph Isomorphism Algorithm for Matching Large Graphs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2004**, *26*, 1367−1372.
(50) Ehrlich, H.-C.; Rarey, M. Systematic Benchmark of Substructure Search in Molecular Graphs. From Ullmann to VF2. *J. Cheminf.* **2012**, *41*, 13.
(51) Schomburg, K.; Ehrlich, H.-C.; Stierand, K.; Rarey, M. From Structure Diagrams to Visual Chemical Patterns. *J. Chem. Inf. Model.* **2010**, *50*, 1529−1535.
(52) Urbaczek, S.; Kolodzik, A.; Fischer, J. R.; Lippert, T.; Heuser, S.; Groth, I.; Schulz-Gasch, T.; Rarey, M. NAOMI: On the Almost Trivial Task of Reading Molecules from Different File Formats. *J. Chem. Inf. Model.* **2011**, *51*, 3199−3207.
(53) *Proasis2*; Desert Scientific Software: Norwest NSW, Australia; http://www.desertsci.com/ (accessed Oct 16, 2012).
(54) Cottrell, S. J.; Olsson, T.; Taylor, R.; Cole, J. C. Validating and Understanding Ring Conformations Using Small Molecule Crystallographic Data. *J. Chem. Inf. Model.* **2012**, *52*, 956−962.
(55) Schulz-Gasch, T.; Schärfer, C.; Guba, W.; Rarey, M. TFD: Torsion Fingerprints as a New Measure To Compare Small Molecule Conformations. *J. Chem. Inf. Model.* **2012**, *52*, 1499−1512.

A6 Maximum common subgraph isomorphism algorithms and their applications in molecular science: A review

Authors:	Hans-Christian Ehrlich, Matthias Rarey
Type of publication:	Journal article
Reference:	Wiley Interdisciplinary Reviews: Computational Molecular Science, 1(1): 68–79, 2011, doi:10.1002/wcms.5
Status:	Published

Advanced Review

Maximum common subgraph isomorphism algorithms and their applications in molecular science: a review

Hans-Christian Ehrlich and Matthias Rarey*

The intuitive description of small and large molecules using graphs has led to an increasing interest in the application of graph concepts for describing, analyzing, and comparing small molecules as well as proteins. Graph theory is a well-studied field and many applications are present in various scientific disciplines. Recent literature describes a number of successful applications to biological problems. One of the most applied concepts aims at finding a maximal common subgraph (MCS) isomorphism between two graphs. We review exact MCS algorithms, especially designed for graphs obtained from small and large molecules, and give an overview of their successful applications. © 2011 John Wiley & Sons, Ltd. *WIREs Comput Mol Sci* 2011 1 68–79 DOI: 10.1002/wcms.5

INTRODUCTION

Chemical database systems are challenged with the task of managing a rising number of molecular entries.[1,2] Especially, the fast and efficient storage and retrieval of the database entries must be ensured. This requires a molecular description based on a sophisticated chemical model. Depending on the chemical question to be addressed, different molecular representations ranging from simple descriptions of physicochemical properties[3] over binary fingerprints[4,5] to graphs and reduced graphs[6–8] are available. Modeling molecules as labeled graphs have a long tradition[9] and is a prerequisite for most modern cheminformatic methods. The representation of molecules by graphs has two major advantages: Graphs are a very intuitive molecular representation close to our elementary chemical understanding, and they form a solid theoretical basis for computer-aided processing. Furthermore, graphs enable a database retrieval via graph isomorphism techniques, i.e., comparing molecules becomes equivalent to comparing labeled graphs. This review focuses on molecular graph comparison techniques, especially addressing the MCS problem. We introduce the graph theoretical background and summarize algorithms solving the MCS problem. Finally, we provide an overview of scientific applications that utilize MCS algorithms.

*Correspondence to: rarey@zbh.uni-hamburg.de

Center for Bioinformatics, Computational Molecular Design, Hamburg, Germany

DOI: 10.1002/wcms.5

PRELIMINARIES

Around 1860, Kekule introduced a structural formula, which is the foundation of modern chemistry. The structural formula is a graph-like representation of molecules commonly used to formulate and exchange chemical knowledge. The formula allows chemists to visualize molecules and quickly identify communalities and differences between them.

Graph Theoretical Background

A graph G is a pair $(V, E)$of vertices and edges. Each edge $e \in E$ connects two adjacent vertices $(v_1, v_2) \in V$. In a labeled graph, vertices (and/or edges) hold arbitrary labels. A graph is simple if each of its edges is undirected and unweighted. Undirected edges have no orientation between the vertices they connect. Unweighted edges have a uniform weight assigned to them. In the following, we only consider simple graphs, except when stated otherwise.

Two graphs G_1 and G_2 are isomorphic if there exists a bijective (one-to-one) mapping between the vertices of G_1 and G_2 such that two vertices in G_1 are connected by an edge, if and only if the corresponding images in G_2 are connected. An induced subgraph

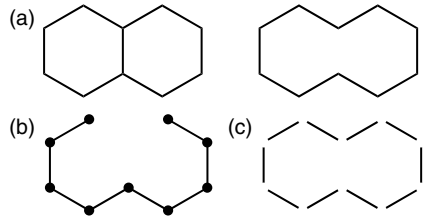


FIGURE 1 | Maximal common induced subgraph (MCIS) versus maximal common edge subgraph (MCES). (a) Molecular graph of decalin (labels not shown). (b) Molecular graph of cyclodecane (labels not shown). (c) MCIS of (a) and (b). (d) MCES of (a) and (b).

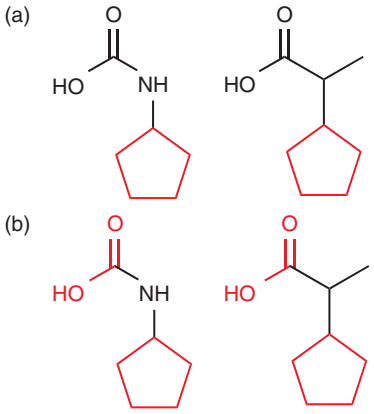


FIGURE 2 | Connected versus disconnected maximal common subgraph (MCS). (a) Connected MCS (red). (b) Disconnected MCS (red).

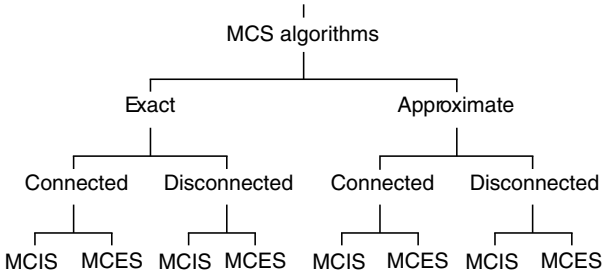


FIGURE 3 | Classification of maximal common subgraph (MCS) algorithms.

of G consists of a subset of vertices $V' \subset V$ and the subset of all edges $E' \subset E$ connecting vertices in V'. An induced subgraph isomorphism exists if G_1 is a subgraph of G_2 (or vice versa), i.e., G_1 is contained in G_2. Finally, a common induced subgraph of two graphs G_1 and G_2 is a graph G_{12} that is isomorphic to a subgraph of G_1 and a subgraph of G_2. Although there are possibly many common subgraphs between two graphs, we will focus on the largest common induced subgraph or maximal common induced subgraph (MCIS). Related to the MCIS is the maximal common edge subgraph (MCES). The MCES is a subgraph with the maximal number of edges common to both G_1 and G_2. Figure 1 shows the difference between MCIS and MCES. Note that the MCIS as well as the MCES of two graphs is not necessarily unique. We will use the term MCS to refer to both, the MCIS as well as the MCES.

Both MCS types can be connected or disconnected. In a connected MCS, each vertex is reachable from every other vertex by a path through the MCS. A disconnected MCS is composed of two or more disconnected components. Figure 2 illustrates the connected and disconnected MCS for the same molecular graph.

The complete MCS algorithm classification scheme is illustrated in Figure 3.

Molecular Representation and Comparison

It is quite obvious that the atoms of a molecule can be easily represented by vertices and bonds by edges. The resulting molecular graph is often labeled to account for atom and bond properties. The degree or number of edges a vertex can have is limited by the number of covalent bonds an atom can form. Therefore, the number of edges linearly depends on the number of atoms. To represent the orientation of atoms in space, it is possible to add three-dimensional (3D) information to a molecular graph.[10,11] However, the graphs considered herein refer to the molecular topology only, except when stated otherwise.

Molecules are considered equal if a one-to-one mapping between all atoms and bonds exists, i.e., the two molecular graphs are isomorphic. To map a pair of atoms or bonds, their labels must be identical. In the case that two molecules are not exactly the same, one molecule can be a substructure of the other. Then, a subgraph isomorphism between the two molecular graphs exists. Alternatively, two molecules share a common substructure, and therefore the molecular graphs share a common subgraph.

Some problems arise when using molecular graphs. In mesomeric structures, e.g., aromatic compounds, different bond localization result in nonisomorphic labeled graphs, although the structures represent the same molecule. For stereoisomeric molecules, additional information, e.g., the relative arrangement of bonds, must be annotated to differentiate between them. Moreover, as molecules exist in potentially many tautomeric forms, the construction of their molecular graphs in a standardized form becomes especially challenging.

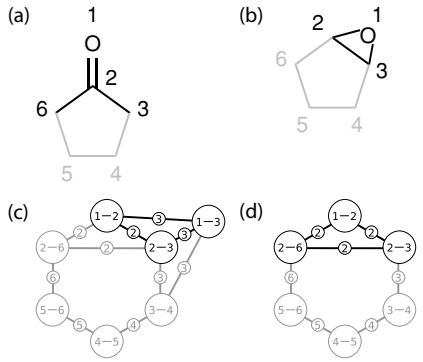


FIGURE 4 | Line graphs. Panels (a) and (b) show the molecular graph of cyclopentanone and epoxycylopentane. The highlighted subgraphs in (a) and (b) become topological equivalent in the corresponding line graphs (c) and (d). A differentiation is only possible by their vertex and edge labels.

Relation between MCIS and MCES

Most of the algorithms described in literature calculate the MCIS and only a few calculate the MCES directly. However, Whitney[12] proved that in cases without trinode/triangle subgraphs, an MCIS can be converted into an MCES. First, the molecular graphs are converted into so-called line graphs that represent the adjacency between edges, which can then in turn be converted in a compatibility graph. Figure 4 shows two molecular graphs: the corresponding line graphs and a trinode/triangle example. The details, especially the trinode/triangle problem, are discussed by Raymond and Willett.[13]

ALGORITHMS

The problem of computing an MCS between two graphs is NP-hard,[14] meaning that no polynomial time algorithm exists (unless P = NP). Nevertheless, many attempts to obtain algorithms useful in practice have been made and most of them are present in the field of computer vision and image recognition.[15] Here, we focus on recent MCS algorithms in the field of molecular science.

To obtain a clear classification of MCS algorithms, we adopt the scheme from Raymond and Willett[13] that differentiates between algorithms calculating exact or approximate, connected or disconnected, vertex-based (MCIS) or edge-based (MCES) solutions. Unfortunately, some published algorithms are not described in enough detail for a clear classification. Especially, the term MCS is often used as a synonym for both, the MCIS and MCES. Therefore, the algorithmic description leaves room for interpretation and an adequate classification becomes difficult.

Maximal Clique-based Algorithms

Calculation of an MCS between two graphs can be reduced to the problem of finding the maximal clique in a compatibility graph. A clique of a graph is a complete subgraph in which each vertex is directly connected to every other vertex. A maximal clique is, therefore, a complete subgraph with the largest possible number of vertices. Note that a graph can incorporate more than one maximal clique. A compatibility graph, also known as association graph,[16,17] modular product graph,[18] or correspondence graph,[19] of two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is defined as the vertex set $V_1 \times V_2$ in which two vertices (v_1, v_2) and (u_1, u_2) are adjacent, if and only if $(v_1u_1) \in E_1$and $(v_2u_2) \in E_2$ or $(v_1u_1) \notin E_1$ and $(v_2u_2) \notin E_2$. For molecular graphs, the compatibility between two vertices or edges is additionally restricted by their labels. The labels must agree according to some compatibility criteria, e.g., the same atom types or bond orders. A maximal clique of a compatibility graph corresponds to the MCIS of the two original graphs. Figure 5 shows two molecular graphs: the compatibility graph and the correspondence between the maximal clique of the compatibility graph and the MCIS.

The approach to reduce the MCS problem to the maximal clique problem is already known for some time[18,20,21] and one of the first applications to chemical structures is described by Kuhl et al.[22] and Brint and Willett.[19] The literature describes many different clique-detection algorithms,[23–27] and Gardiner et al.[28] analyzed the performance of the most common ones. Two widely used method for arbitrary graphs are the algorithms by Bron and Kerbosch[23] and Carraghan and Pardalos.[25] Although chemical graphs are in general sparse, their compatibility graphs tend to be dense. Therefore, the general clique-detection algorithms, which do not use any chemical

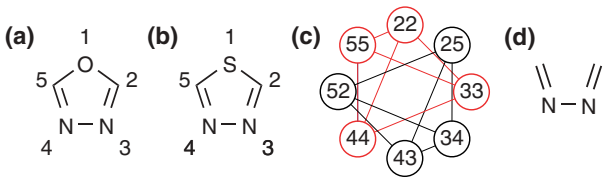


FIGURE 5 | Maximal clique in a compatibility graph. (a) Molecular graph of 1,3,4-oxadiazole. (b) Molecular graph of 1,3,4-thiadiazole. (c) Compatibility graph of (a) and (b). The graph has two maximal cliques indicated with red and black lines. (d) Maximal common subgraph (MCS) of 1,3,4-oxadiazole and 1,3,4-thiadiazole. Both cliques resemble the same MCS.

information, are often slow. Raymond et al.[29,30] developed rapid similarity calculation (RASCAL), a branch-and-bound procedure that uses multiple chemically motivated heuristic strategies to improve the efficiency of clique detection on molecular graphs. RASCAL calculates the exact, disconnected MCES by converting the MCES problem into the MCIS problem.[12] In RASCAL, the two input graphs are transformed into line graphs from which the compatibility graph is constructed. A maximal clique in that compatibility graph corresponds to an MCES because each vertex resembles an edge. The RASCAL procedure was adapted to calculate the exact, disconnected MCIS and MCES on a reduced version of molecular graphs.[31] In a reduced molecular graph, a vertex no longer resembles a single atom but rather a group of atoms, e.g., a functional group. A recent developed branch-and-bound method[32] based on the clique-detection method of Carraghan and Pardalos[25] is described to detect all exact MCESs. Another branch-and-bound clique-detection algorithm[33] is an extension of Mehlhorn's algorithm[34] and uses a compatibility graph with a weakened edge definition such that two vertices (v_1, v_2) and (u_1, u_2) are adjacent, if and only if $(v_1u_1) \in E_1$ or $(v_1u_1) \notin E_1$ and $(v_2u_2) \notin E_2$. The result is an exact, disconnected MCS, in which two vertices can be adjacent in the first graph and nonadjacent in the second. Therefore, a circular structure can be matched to a linear one.

The computing time of clique-detection algorithms increases exponentially with the number of vertices and edges in the compatibility graph. The above-mentioned concept of chemical labeling, e.g., with atom types[35] or bond orders,[32] reduces the number of vertices and edges substantially. Often, only because of chemical labeling, the MCS calculation becomes feasible for an application in molecular sciences. The methods described so far calculate the disconnected MCS; however, most clique-based MCS algorithms can be modified to calculate the connected MCS.[36,37]

Backtracking Algorithms

An MCS of two graphs is usually represented by a bijective mapping of a subset of vertices of the first graph to a subset of vertices of the second. If this mapping is built-up sequentially vertex by vertex, a tree structure is defined that can be searched by classical backtracking algorithms (Figure 6 shows an example). During the traversal, a current subsolution is gradually extended to guide the search toward the final solution. When an extension does not lead to a valid or better solution, the underlying branch of the search tree gets pruned; therefore, reducing the number of backtracking iterations needed to find the MCS.

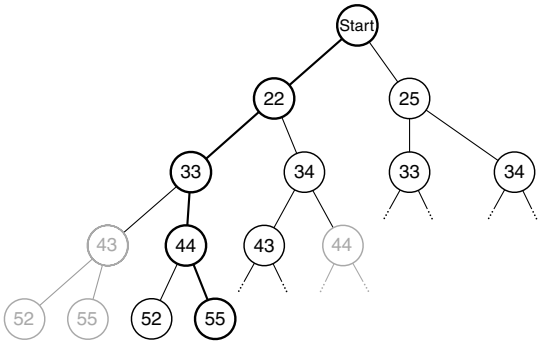


FIGURE 6 | Backtracking search tree for 1,3,4-oxadiazole and 1,3,4-thiadiazole as shown in Figure 5(a) and (b). One solution is highlighted. Cut branches are shown in gray.

Two historical backtracking methods were introduced by McGregor[38] and Ullmann.[39] McGregor[38] appears to be the first who draws a difference between an MCIS and MCES. The Ullmann[39] algorithm calculates subgraph isomorphism rather than an MCS but is worth mentioning because it was the fastest algorithm at its time and is the basis for other subgraph isomorphism algorithms.[40] Additionally, the algorithms and its variations are widely used in today's chemical substructure search systems.[41]

Most recently, Cao et al.[42] presented a backtracking procedure that calculates the exact, disconnected MCIS with control over the number of disconnected components. The algorithm works directly on the molecular graphs and makes use of multiple strategies to prune the search tree. One pruning strategy excludes extensions that exceed the allowed number of disconnected components. Another, the induced subgraph heuristic, is derived from the definition of an induced subgraph and identifies infeasible sets of vertex mappings in which the vertices do not lead to an edge-compatible vertex assignment. To further reduce the search tree, the algorithm applies a branch-and-bound strategy that uses a current suboptimal solution to calculate an estimate of the maximal possible MCIS and apply it as upper bound on the final solution. If the estimate is worse than the best solution found so far, the branch is not further explored. To find large suboptimal solutions first, the vertices are processed in decreasing order according to their number of neighbors to the current common subgraph. Therefore, the next processed vertex mostly improves the upper bound, achieving a further acceleration of the search process.

Berlo et al.[43] extended Cao et al.[42] method to calculate all exact, connected MCESs. Although the induced subgraph heuristic uses the vertex connectivity to identify vertices that do not result in a valid one-to-one mapping, it cannot be applied when calculating an MCES. However, Berlo et al.[43] use a similar vertex ordering scheme and state that their method reduces the search tree by adding multiple edges to a current solution in one step. Unfortunately, it is not proven that simultaneously adding multiple edges always gives an exact MCES. A comparison shows that Cao et al.[42] algorithms are faster up to an MCS of about 20 vertices and is then surpassed by Berlo et al.[43] method.

Dynamic Programming

Dynamic programming (DP)[44] is a long-known mathematical technique for solving multistage decision problems. The central element of DP algorithms is the hierarchical division of problems into subproblems, which are solved bottom-up storing and reusing partial solution, a technique named memorization. DP is most efficiently applied when subproblems can be solved independently from each other. Depending on the structure of subproblems, DP algorithms can achieve polynomial runtime behavior.

Because of NP-hardness of the MCS problem, it is extremely unlikely that a DP scheme results in a polynomial-time MCS algorithm. However, the MCS problem becomes easier to solve for certain graph classes. Most importantly, the MCS between two trees can be calculated in polynomial time using DP.[45] Depending on the application, it is, therefore, worthwhile to carefully analyze the molecular graphs involved. The reduced graph descriptor feature tree[7] makes use of this concept by representing molecules as trees such that an efficient algorithm for calculating the largest common subtree rather than an exponential-time MCS algorithm can be applied.

Schietgat et al.[46] developed a DP algorithm that calculates a so-called block-and-bridge preserving exact, connected MCIS. The algorithm is based on Horvarth et al.[47] and is specially designed for outerplanar graphs. Fortunately, most molecular graphs are outerplanar. When molecules are compared, it is often not desired to assign circular substructures to noncircular ones. Therefore, during the construction of an MCS, the algorithm only matches atoms of biconnected components (blocks) and edges connecting blocks (bridges) to each other. This constraint and the fact that the considered graphs are outerplanar make a polynomial runtime for solving the MCS problem possible.

Multiple MCS Isomorphism

The algorithms described so far always search for an MCS between two graphs. Finding the MCS between multiple graphs is an interesting problem when applied to molecules and has received comparatively low attention in molecular science. Nevertheless, we want to illustrate an example for multiple MCS calculation. In cheminformatics, enumerated subgraphs are frequently used as molecular descriptors. A multiple, connected MCES can be obtained by enumerating all possible subgraphs and extracting the largest one common to all input graphs.[48] The subgraphs are retrieved from the full extended connectivity fingerprint (ECFP),[49] a common molecular descriptor. The ECFP algorithm generates circular-growing substructures using an adaptation of Morgan's algorithms[50] for canonical labeling of molecular graphs. Note that this approach is neither exact nor it can be scaled up for larger graphs.

Benchmarking

Many MCS algorithms have been developed, but very little effort has been made to create a uniform test environment to compare the algorithms with respect to runtime and their field of application. Most often, the algorithms are evaluated in a special experimental setting to show that they can outperform previous ones. The result is a number of evaluations that are hard to compare. Conte et al.[51] face the problem by assembling a diverse set of synthetical graphs that is composed of randomly connected graphs, regular and irregular meshes, and regular and irregular bounded valence graphs. Note that irregular bounded valence graphs have properties similar to molecular graphs. Three classical MCS algorithms were tested with this benchmark set: McGregor's[38] backtracking procedure, and the algorithms of Durand et al.[52] and Balas and Yu,[24] both searching for the maximal clique. None of the three algorithms showed a superior runtime behavior for all kinds of graphs. McGregor's[38] algorithm performs well when graphs are sparse and/or small. However, for irregular graphs with bounded valence, Durand's algorithm performs best. The comparison shows that an appropriate application of MCS algorithms strongly depends on the considered problem.

APPLICATIONS

MCS algorithms have a variety of applications in molecular science and the number is constantly rising. The complete scope of applications can certainly

not be covered by this article. Nevertheless, we want to give a broad overview of areas that use MCS algorithms. In the following, we provide examples for biological problems that are addressed by using MCS algorithms. The first part considers algorithms applied to molecular graphs obtained from small organic molecules such as drugs, whereas the second part provides examples for applications on graphs derived from large molecules such as proteins. The two types of graphs are fundamentally different. A small molecule graph is composed of atoms and bonds or groups of atoms and topological distances. In proteins, complete amino acids and the geometric distances between them are usually used to construct the graph. Also, the interpretation of an MCS differs for small molecules and proteins. An MCS of small molecules is often used as structural similarity measure; whereas in proteins, it resembles a common structure motif. It appears that the algorithms only find little application for ribonucleic acids (RNAs), even though common structural patterns are of major interest when studying RNA.

Small Organic Molecules

MCS algorithms can be applied to identify ligand families, predict ligand activity, or to analyze the mechanism of reactions. Although the most common application of MCS algorithms is to retrieve similarity values, the examples show in detail how small molecules are transformed into graphs and how to retrieve a measure of similarity from an MCS. Other applications of MCS algorithms involve ligand alignment,[53] the determination of quantitative structure–property relationships (QSPR),[54] and pharmacophore modeling.[55,56]

Compound Classification

A central problem when dealing with small molecules in pharmaceutical research is to group individual compounds into structurally related families or clusters. Manually grouping large databases is a tedious task and automated procedures are, therefore, often used. Automated clustering methods need a similarity metric for pairwise comparison of structures and a clustering algorithm for sorting compounds into structurally related groups. An MCS can describe common connected substructures or scaffolds as well as a set of largest common fragments or functional groups between two molecules. Stahl et al.[35] analyzed the usability of different similarity metrics obtained from disconnected MCESs for compound clustering. The motivation to use a disconnected MCES is to detect similarity between compounds that do not share a large common substructure but rather common functional groups that are disconnected. Six different MCES algorithms (Rambin,[57] an implementation of the Bron–Kerbosch algorithm,[23] Dfmax and Nmclique,[58] Pardalos,[59] Wood,[60] and Rascal[29]) and a variety of clustering methods were compared on a set of 466 compounds known to bind to nine different targets. From the number of vertices and edges that comprise the MCS, a similarity between two molecules is calculated according to[29] the following equation:

$$\mathrm{sim}(A, B) = \frac{(|V(\mathrm{MCS}_{A,B})| + |E(\mathrm{MCS}_{A,B})|)}{(|V(A)| + |V(B)|)(|E(A)| + |E(B)|)} \quad (1)$$

where $\mathrm{MCS}_{A,B}$ is the MCS between molecules A and B.

The similarity calculation is extended by two correction terms. The first penalizes a different relative topological arrangement between the three largest functional groups. The second raises the similarity index if the largest MCES fragment comprises more than 70% of one molecule indicating a common scaffold. A combination of the RASCAL–MCES algorithm with the average linkage unweighted pair group method with arithmetic mean (UPGMA)[61] cluster method most accurately separates compounds into their distinct classes while creating relatively pure clusters and the least number of singletons.

Compound Activity Prediction

A key step in finding new drugs is the identification of chemical compounds that shows activity in specific biological processes. Virtual screening techniques try to identify potential active compounds by large-scale *in silico* activity prediction with the aim to reduce the number of molecules that need to be experimentally tested. The similarity principle in drug design states that, statistically, structurally similar compounds tend to have similar activity.[62] Rarey and Dixon[7] identify molecules that belong to the same class of inhibitors on a set of 972 molecules[63] from the MACCS Drug Data Report database[64] and predict binding geometries on 58 molecules[65] taken from the Brookhaven Protein Data Bank.[66] Molecules are reduced to acyclic graphs, feature trees, in which vertices represent functional groups and edges resemble the relative topological arrangement of groups. A vertex describes steric features, e.g., van der Waals volume, and chemical features such as possible interactions a group can form with a potential receptor. A weighted average over the similarity values of vertex matches in the exact, disconnected MCIS of two feature trees is used as the similarity between two

molecules. Predictions of ligand-binding geometry have an average root mean square derivation (RMSD) under 4 Å in 61% of the 58 molecules. In a virtual screening experiment, enrichment factors at 1% of the 972 molecules screened are similar when using feature trees and daylight fingerprints. Nevertheless, 50% of the top-ranking molecules obtained using feature trees differ from the ones found using daylight fingerprints.

Schietgat et al.[46] successfully perform a compound activity prediction on the National Cancer Institute (NCI) dataset containing about 70,000 active and inactive compounds to treat human tumor cells. The molecules are transformed into binary strings that encode the occurrence of frequent substructure patterns in a set of active compounds. A substructure mining algorithm[47] in combination with the block-and-bridge preserving MCIS algorithm (see above) calculates all patterns present in a ligand, and the presence of a most frequent pattern is indicated by setting the correspond bit. Finally, a support vector machine classifies the compounds based on their binary description. A 10-fold cross-validation shows a prediction performance comparable to other methods based on subgraph isomorphism,[67] fingerprints, or kernels.

Scaffold Hopping

A problem of similarity-based methods for virtual screening is their tendency to only identify compounds that are structurally very similar to the original active molecules. However, it is of special pharmaceutical interest to find novel molecules that are built from different molecular scaffolds while preserving activity against the same target protein. Different scaffold hopping methods successfully address this task.[68] Barker et al.[31] investigate the scaffold-hopping ability of different MCS-based molecular similarity measures. The molecular graph is transformed into a reduced graph with vertices resembling functional groups and edges representing the topological distances between them. A reduced graph is similar to a pharmacophoric description, which describes the structural features essential for the biological activity. Barker et al.[31] adapted the similarity formula from Eq. (1) and studied the influence on the similarity value when using either an MCIS or an MCES between two reduced graphs. A comparison with daylight fingerprints[4] in a simulated virtual screening experiment on a filtered version of the MDL Drug Data Report indicates a similar enrichment ability at 1% of the database screened and only small differences when using the MCIS or MCES, respectively. The method retrieves about the same number of unique scaffolds, but the scaffolds are complementary in terms of diversity to those found using daylight fingerprints.

Reaction Mapping

Understanding the mechanism of enzymatically catalyzed reactions is of major interest when studying metabolic pathways of the cell. A chemical reaction transforms the reactant molecule to the product by deleting existing bonds and forming new ones. These reaction centers can be experimentally identified but only in a small scale. The works of Korner and coworkers[69,70] are aimed at automatically determining reaction centers in high-throughput applications. Each reactant and product is modeled as a molecular graph in which vertices are atoms and edges are bonds. An edge also holds a weight that corresponds to the stability of the bond. A weighted MCES (wMCES) that maximizes the sum of common edge weights is used to determine the set of bonds that are most likely conserved when a reactant is changed to the product. All bonds not part of the wMCES are either broken or formed during the reaction. The sets of conserved bonds and reaction bonds allow an identification of a bijective atomic mapping between reactant and product. For the experiment, the RASCAL algorithm was modified to calculate the wMCES and its automated application most often results in a correct mapping of over 8000 manually mapped chemical reactions obtained from Kyoto Encyclopedia of Genes and Genomes (KEGG)[71] and BioPath database.[72]

Quantitative Structure–Activity Relationships

The recently exponential growth in the number of publications presenting quantitative structure–activity relationship (QSAR) and QSPR shows the importance of an accurate prediction of compound activity/property in modern chemistry and biochemistry. The concept of QSAR/QSPR is to transform chemical knowledge and intuition into mathematically derived equations that correlate the structure with a known activity/property. With such a model, it is possible to search any number of compounds, even the ones that are not yet synthesized, for the desired activity/property. Cuadrado et al.[73] derive a QSAR model to predict the blood–brain barrier permeability from a known set of 136 active compounds. The QSAR model describes each ligand as a vector of similarity values against all actives. In principle, any measure that describes the similarity between two ligands can be used. For a detailed review on molecular similarity measures, see Refs 74 and 75. Cuadrado et al.[73] derive similarity values by approximating the

van der Waals surface area[76] of an extended MCIS (EMCIS). The EMCIS contains information about the position of substituents that are not part of the original MCIS. The model is trained and tested using leave-one-out validation to guarantee a high prediction performance. An independent test indicates a prediction performance similar to previous QSAR models and other approaches based on 3D methods or neural networks.

Ribonucleic Acid

Structure comparison is one of the central tools used for function prediction of novel RNAs. Often, a sequence comparison is sufficient for finding related RNAs with known function from which a function prediction can be obtained. For some major RNA families, such as transfer RNA and ribosomal RNA, sequence comparison fails due to the low-sequence similarity between family members. Fortunately, these families show a highly conserved fold. Therefore, a direct comparison of the secondary structure can reveal similarities not present on the sequence level. Chao[32] compared RNA structures of different complexity and searched for the presents of iron response elements (IREs) in the untranslated region of human messenger RNA (mRNA). The mRNA structure is modeled as a graph, with nucleotides forming the vertices and edges resembling either covalent bonds or hydrogen contacts between nucleotides. The search for IREs in human mRNAs yield 26 genes from which six are known to contain IREs. The comparison of structures within different RNA families results in the proposal of an extended vertex-encoding scheme. Instead of labeling each vertex with the corresponding nucleotide symbol, vertices are labeled according to their secondary structure. The scheme is useful when only the RNA structure, regardless of its sequence, should be compared.

Proteins

The two main application fields of MCS algorithms when studying proteins are protein alignment that identifies global structural similarities between proteins and pattern analysis in which the major interest lies in local similarities.

Structural Alignment

Understanding the function and architecture of proteins is a central problem in molecular biology. The 3D protein fold mainly determines the protein function, stability, and general behavior. Therefore, the structural comparison of proteins can give valuable insights into the nature of proteins. Jain and Lappe[77] compare protein structures by approximately solving the contact map overlap (CMO) problem[78] in which a protein structure is modeled as a contact graph and the MCS of two proteins describes the similarity between them. A contact graph consists of protein residues and two residues are connected by an edge if their distance in space is small enough. To obtain a solution to the CMO problem, the approximate MCES algorithm softassign[79,80] maximizes the number of common contacts between two proteins, and a self-developed DP strategy ensures that the order of residues forming the solution is the same in both proteins. The algorithm computes almost optimal matches on a CMO test set compiled by Strickland et al.[81] and shows running times around minutes for proteins up to 1500 residues in size. The results indicate that the method is faster than other CMO algorithms, the runtime scales well with increasing protein size, and that the algorithm is most efficiently applied when comparing structurally similar proteins.

Structural Pattern Analysis

The identification of substructures or motifs in proteins that are related to a specific function or fold generally leads to a hypothesis about the evolutionary origin or conducted function of the protein. The analysis of complete databases for frequently occurring motifs is an opportunity to identify conserved substructures. Caboche et al.[33] analyzed the NORINE database[82] for structural commonalities between nonribosomal peptides (NRPs). In contrast to regular proteins, their structure can be partially or fully cyclic, branched, or even polycyclic. The NORINE database provides about 700 NRPs as molecular graphs. A vertex of a graph corresponds to a monomer and an edge to a chemical bond between two monomers. The database is successfully analyzed for family-specific structural features in NRPs, and an example application is given to predict the product of NRP-producing proteins from the protein structure.

Artymiuk et al.[83] study common folding motifs between adenylyl cyclase and DNA polymerase 1 and between biotin carboxylase and adenosine diphosphate (ADP)-forming peptide synthetases in detail. The molecular graph representation of the analyzed proteins makes use of the fact that the spatial arrangement of secondary structure elements (SSEs) describes the protein's fold. Although SSEs are approximately linear structures, they are modeled by a vector drawn along their major axis. A molecular graph representing the protein structure is then composed such that each vertex holds an SSE vector and each edge describes a geometric relationship

between a pair of them. The Bron and Kerbosch[23] algorithm is modified to account for edge labels when searching for the maximal clique. The resulting disconnected MCIS gives the structural relationship between two protein folds. Artymiuk et al.[83] revealed common folding modes between the proteins that indicate similar function between adenylyl cyclase and DNA polymerase 1 and homology between the families of biotin carboxylase and ADP-forming peptide synthetases.

CONCLUSION

The aim of this review is to provide an overview of current algorithms that solve the MCS problem for molecular graphs and to show their general applications in the field of molecular science. Most algorithms address MCS problem by solving the maximal clique problem on a compatibility graph. However, two of the most recently presented algorithms[42,43] are backtracking procedures. Because of its good performance on molecular graphs, RASCAL based on clique detection belongs to the widely applied implementations.

One major application of MCS algorithms is their use to determine similarity between small organic compounds. In contrast to fingerprint methods, the MCS captures topological relations between atoms or functional groups. It, therefore, results in a similarity concept well reflecting the synthetic chemists understanding of molecular relationships. The additional topological information can be of high relevance when searching for alternative ligands with similar biological activity. Applied to proteins, MCS algorithms can accomplish the detection of global and local similarities.

The special kind of MCS used to address a problem is of central importance for the application as well as for algorithm design. We hope that future publications use the proposed classification scheme and give a clear description of the algorithm's intended application field. New MCS algorithms need to be compared with existing methods in a reproducible environment, preferably on a generalized test set or at least on a large number of varying graphs. The number of test sets available, especially those that resemble the properties of molecular graphs, is small; therefore, we encourage further research.

Because of NP-hardness, algorithms solving the MCS problem in general will likely stay exponential in runtime requirement. Nevertheless, for a specific application, there are typically lots of options to optimize. For performance, modeling vertex and edge compatibility is critical. Moreover, some graph classes such as trees and planar graphs allow much faster methods for MCS calculation. For future applications of MCS in molecular science, time spent for carefully modeling the problem as an MCS problem is, therefore, well invested.

We hope that this review provides an entry point into the current state of MCS algorithms and gives an insight into the broad range of applications in molecular science. A number of problems in molecular science can be solved with MCS-based approaches and, therefore, we encourage exploring new fields for their broader application.

REFERENCES

1. Irwin JJ, Shoichet BK. ZINC—a free database of commercially available compounds for virtual screening. *J Chem Inf Model* 2005, 45:177–182.
2. Geer LY, Marchler-Bauer A, Geer RC, Han L, He J, He S, Liu C, Shi W, Bryant SH. The NCBI Biosystems database. *Nucleic Acids Res* 2009, 38:492–496.
3. Xue L, Godden JW, Bajorath J. Evaluation of descriptors and mini-fingerprints for the identification of molecules with similar activity. *J Cheml Inf Comput Sci* 2000, 40:1227–1234.
4. *Daylight Theory Manual*. Daylight Chemical Information Systems Inc.
5. Durant JL, Leland BA, Henry DR, Nourse JG. Reoptimization of MDL keys for use in drug discovery. *J Chem Inf Comput Sci* 2002, 42:1273–1280.
6. Takahashi Y, Sukekawa M, Sasaki S. Automatic identification of molecular similarity using reduced-graph representation of chemical structure. *J Chem Inf Comput Sci* 1992, 32:639–643.
7. Rarey M, Dixon JS. Feature trees: a new molecular similarity measure based on tree matching. *J Comput-Aided Mol Des* 1998, 12:471–490.
8. Harper G, Bravi GS, Pickett SD, Hussain J, Green DVS. The reduced graph descriptor in virtual screening and data-driven clustering of high-throughput screening data. *J Chem Inf Comput Sci* 2004, 44:2145–2156.
9. Trinajstic N. *Chemical Graph Theory*. 2nd ed. New Directions in Civil Engineering. CRC Press; 1992.
10. Gund P. Three-dimensional pharmacophoric pattern searching. *Prog Mol Subcell Biol* 1977, 5:117–143.

11. Gund P. Pharmacophoric pattern searching and receptor mapping. *Ann Rep Med Chem* 1979, 14:299–308.
12. Whitney H. Congruent graphs and the connectivity of graphs. *Am J Math* 1932, 54:150–168.
13. Raymond JW, Willett P. Maximum common subgraph isomorphism algorithms for the matching of chemical structures. *J Comput-Aided Mol Des* 2002, 16:521–533.
14. Garey MR. *Computers and Intractability*. New York: W. H. Freeman and Company; 1979.
15. Bunke H. Graph Matching: theoretical foundations, algorithms, and applications In: *International Conference on Vision Interface*. Quebec, Canada: Montreal; 2000, 82–88.
16. Pelillo M, Siddiqi K, Zucker SW. Matching hierarchical structures using association graphs. *IEEE Trans Pattern Anal Machine Intell* 1999, 21:1105–1120.
17. Yang B, Snyder WE, Bilbro GL. Matching oversegmented 3D images to models using association graphs. *Image Vis Comput* 1989, 7:135–143.
18. Barrow HG, Burstall RM. Subgraph isomorphism, matching relational structures and maximal cliques. *Inf Process Lett* 1976, 4:83–84.
19. Brint A, Willett P. Algorithms for the identification of 3-dimensional maximal common substructures. *J Chem Inf Comput Sci* 1987, 27:152–158.
20. Levi G. A note on the derivation of maximal common subgraphs of two directed or undirected graphs. *Calcolo* 1973, 9:341–352.
21. Cone M, Venkataraghavan R, McLafferty F. Molecular structure comparison program for the identification of maximal common substructures. *J Am Chem Soc* 1977, 99:7668–7671.
22. Kuhl F, Crippen G, Friesen D. A combinatorial algorithm for calculating ligand-binding. *J Comput Chem* 1984, 5:24–34.
23. Bron C, Kerbosch J. Finding all cliques of an undirected graph. *Commun of the ACM* 1973, 16:575–577.
24. Balas E, Yu CS. Finding a maximum clique in an arbitrary graph. *SIAM J Comput* 1986, 15:1054–1068.
25. Carraghan R, Pardalos P. An exact algorithm for the maximum clique problem. *Oper Res Lett* 1990, 9:375–382.
26. Shindo M, Tomita E. A simple algorithm for finding a maximum clique and its worst-case time complexity. *Syst Comput Japan* 1990, 21:1–13.
27. Babel L. Finding maximum cliques in arbitrary and in special graphs. *Computing* 1991, 46:321–341.
28. Gardiner EJ, Artymiuk PJ, Willett P. Clique-detection algorithms for matching three-dimensional molecular structures. *J Mol Graph Model* 1997, 15:245–253.
29. Raymond JW, Gardiner EJ, Willett P. RASCAL: calculation of graph similarity using maximum common edge subgraphs. *The Computer Journal* 2002, 45:631–644.
30. Raymond JW, Gardiner EJ, Willett P. Heuristics for similarity searching of chemical graphs using a maximum common edge subgraph algorithm. *J Chem Inf Comput Sc* 2002, 42:305–316.
31. Barker EJ, Buttar D, Cosgrove DA, Gardiner EJ, Kitts P, Willett P, Gillet VJ. Scaffold hopping using clique detection applied to reduced graphs. *J Chem Inf Model* 2006, 46:503–511.
32. Chao S-Y. Maximum common substructure extraction in RNA secondary structures using clique detection approach. *World Acad Sci, Eng Technol* 2008, 45:219–228.
33. Caboche S, Pupin M, Leclere V, Jacques P, Kucherov G. Structural pattern matching of nonribosomal peptides. *BMC Struct Biol* 2009, 9:15.
34. Mehlhorn K. Data structures and algorithms 2: graph algorithms and NP-completeness. In: *Monographs in Theoretical Computer Science. An EATCS Series*. Vol. 2. Springer; London, UK 1984.
35. Stahl M, Mauser H, Tsui M, Taylor NR. A robust clustering method for chemical structures. *J Med Chem* 2005, 48:4358–4366.
36. Jauffret P, Tonnelier C, Hanser T, Kaufmann G. Machine learning of generic reactions: 2. toward an advanced computer representation of chemical reactions. *Tetrahedron Comput Methodol* 1990, 3:335–349.
37. Koch I. Enumerating all connected maximal common subgraphs in two graphs. *Theor Comput Sci* 2001, 250:1–30.
38. McGregor JJ. Backtrack search algorithms and the maximal common subgraph problem. *Softw: Pract Exp* 1982, 12:23–34.
39. Ullmann JR. An algorithm for subgraph isomorphism. *J Assoc Comput Machinery* 1976, 23:31–42.
40. Wong AKC, Akinniyi FA. An algorithm for the largest common subgraph isomorphism using the implicit net. *Proc IEEE Syst, Man, and Cybern* 1983, 1:197–201.
41. Barnard J. Substructure searching methods — old and new. *Journal of Chem Inf Comput Sci* 1993, 33:532–538.
42. Cao Y, Jiang T, Girke T. A maximum common substructure-based algorithm for searching and predicting drug-like compounds. *Bioinformatics* 2008, 24:366–374.
43. Berlo RJPv, Groot MJLd, Reinders MJT, Ridder Dd. Efficient calculation of compound similarity based on maximum common subgraphs and its application to prediction of gene transcript levels. In: *Information & Communication Theory Group, Technical Report.Delft, the Netherlands: Delft University of Technology*; 2009.
44. Cormen TH, Leiserson CE, Rivest RL, Stein C. *Introduction to Algorithms*. Cambridge, MA: MIT Press; 2001.

45. Gupta A, Nishimura N. Finding largest subtrees and smallest supertrees. *Algorithmica* 1998, 21:183–210.
46. Schietgat L, Ramon J, Bruynooghe M, Blockeel H. An efficiently computable graph-based metric for the classification of small molecules. In: *Proceedings of the 11th International Conference on Discovery Science*. Berlin, Heidelberg: Springer-Verlag; 2008, 197–209.
47. Horvarth T, Ramon J, Wrobel S. Frequent subgraph mining in outerplanar graphs. In: *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY: ACM; 2006, 197–206.
48. Hassan M, Brown RD, Varma-O'brien S, Rogers D. Cheminformatics analysis and learning in a data pipelining environment. *Mol Divers* 2006, 10:283–299.
49. SciTegic. *Pipeline Pilot — Basic Chemistry Collection User Guide*. Telesis Court, San Diego, CA: ; 2006, 92121–4779.
50. Morgan HL. The generation of a unique machine description for chemical structures—a technique developed at chemical abstracts service. *J Chem Doc* 1965, 5:107–113.
51. Conte D, Guidobaldi C, Sansone C. A comparison of three maximum common subgraph algorithms on a large database of labeled graphs. In: *Graph Based Representations In Pattern Recognition*. Berlin: Springer-Verlag; 2003, 130–141.
52. Durand PJ, Pasari R, Baker JW, Tsai C-C. An efficient algorithm for similarity analysis of molecules. *Internet J Chem* 1999, 2.
53. Thorner DA, Willett P, Wright PM, Taylor R. Similarity searching in files of three-dimensional chemical structures: representation and searching of molecular electrostatic potentials using field-graphs. *J Comput-Aided Mol Des* 1997, 11:163–174.
54. Cuissart B, Touffet F, Cremilleux B, Bureau R, Rault S. The maximum common substructure as a molecular depiction in a supervised classification context: experiments in quantitative structure/biodegradability relationships. *J Chem Inf Comput Sci* 2002, 42:1043–1052.
55. Martin YC, Bures MG, Danaher EA, DeLazzer J, Lico I, Pavlik PA. A fast new approach to pharmacophore mapping and its application to dopaminergic and benzodiazepine agonists. *Jf Comput-Aided Mol Des* 1993, 7:83–102.
56. Wolber G, Seidel T, Bendix F, Langer T. Molecule-pharmacophore superpositioning and pattern matching in computational drug design. *Drug Discov Today* 2008, 13:23–29.
57. Available at: http://www.twisted-helices.com/computing/rambin/rambin.html (Accessed November 11, 2010).
58. Available at: ftp://dimacs.rutgers.edu/pub/challenge/graph/solvers/ (Accessed November 11, 2010).
59. Pardalos PM, Xue J. The maximum clique problem. *J Glob Opt* 1992, 4:301–308.
60. Wood DR. An algorithm for finding a maximum clique in a graph. *Oper Res Lett* 1997, 21:211–217.
61. Sokal RR, Michener CD. A statistical method for evaluating systematic relationships. *Univ Kans Sci Bull* 1958, 28:1409–1438.
62. Johnson EG, Maggiora GM, *Concepts and Applications of Molecular Similarity*. New York: John Wiley & Sons; 1990.
63. Briem H, Kuntz ID. Molecular similarity based on DOCK-generated fingerprints. *J Med Chem* 1996, 39:3401–3408.
64. *MACCS Drug Data Report (MDDR)*. San Leandro, CA: MDL Information Systems Inc.
65. Lemmen C, Lengauer T, Klebe G. FLEXS: a method for fast flexible ligand superposition. *J Med Chem* 1998, 41:4502–4520.
66. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN, Bourne PE. The protein data bank. *Nucleic Acids Res* 2000, 28:235–242.
67. Bringmann B. Don'be afraid of simpler patterns. In: 10th *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*. Berlin: Springer; 2006, 55–66.
68. Schneider G, Schneider P, Renner S. Scaffold-hopping: how far can you jump? *QSAR Comb Sci* 2006, 25:1162–1171.
69. Korner R, Apostolakis J. Automatic determination of reaction mappings and reaction center information. 1. the imaginary transition state energy approach. *J Chem Inf Model* 2008, 48:1181–1189.
70. Apostolakis J, Sacher O, Korner R, Gasteiger J. Automatic determination of reaction mappings and reaction center information. 2. validation on a biochemical reaction database. *J Chem Inf Model* 2008, 48:1190–1198.
71. Kanehisa M. The KEGG database. *Novartis Foundation Symp* 2002, 247:91–101; discussion 101–103, 119–128, 244–252.
72. Reitz M, Sacher O, Tarkhov A, Trumbach D, Gasteiger J. Enabling the exploration of biochemical pathways. *Org Biomol Chem* 2004, 2:3226–3237.
73. Cuadrado MU, Ruiz IL, Gomez-Nieto MA. QSAR models based on isomorphic and nonisomorphic data fusion for predicting the blood brain barrier permeability. *J Comput Chem* 2007, 28:1252–1260.
74. Maggiora GM, Shanmugasundaram V. Molecular similarity measures. *Methods in Mol Biol (Clifton, N.J.)* 2004, 275:1–50.

75. Willett P, Barnard JM, Downs GM. Chemical similarity searching. *J Chem Inf Comput Sci* 1998, 38:983–996.
76. Gutman I, Kortvelyesi T. Wiener indices and molecular surfaces. *Zeitschrift fur Naturforschung* 1995, 50a:669–671.
77. Jain BJ, Lappe M. Joining softassign and dynamic programming for the contact map overlap problem. In proceedings of the 1st international conference on Bioinformatics research and development. *BIRD* 2007. Berlin: Springer Heidelberg, 410–424.
78. Godzik A, Skolnick J. Flexible algorithm for direct multiple alignment of protein structures and sequences. *Comput Appl Biosci : CABIOS* 1994, 10:587–596.
79. Gold S, Rangarajan A. A graduated assignment algorithm for graph matching. *Pattern Anal and Machine Intell, IEEE, Trans on Pattern Anal and Machine Intell*, 1996, 18:377–388.
80. Ishii S, Sato MA. Doubly constrained network for combinatorial optimization. *Neurocomputing* 2002, 43:239–257.
81. Strickland DM, Barnes E, Sokol JS. Optimal protein structure alignment using maximum cliques. *Oper Res* 2005, 53:389–402.
82. Caboche S, Pupin M, Leclere V, Fontaine A, Jacques P, Kucherov G. NORINE: a database of nonribosomal peptides. *Nucleic Acids Res* 2008, 36:326–331.
83. Artymiuk P, Spriggs R, Willett P. Graph theoretic methods for the analysis of structural relationships in biological macromolecules. *J Am Soc Inf Sci Technol* 2005, 56:518–528.

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbständig und ohne Benutzung anderer als der angegeben 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.

Hamburg, der 10. Oktober 2013

(Hans-Christian Ehrlich)