JCIM JOURNAL OF CHEMICAL INFORMATION AND MODELING

Cite This: J. Chem. Inf. Model. 2019, 59, 2560−2571

Article

pubs.acs.org/jcim

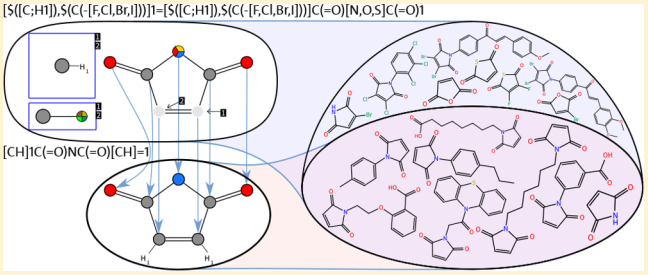
Comparing Molecular Patterns Using the Example of SMARTS: Theory and Algorithms

Robert Schmidt, Emanuel S. R. Ehmki, Farina Ohm,[†] Hans-Christian Ehrlich,[‡] Andriy Mashychev,[§] and Matthias Rarey*

ZBH - Center for Bioinformatics, Bundesstraße 43, 20146 Hamburg, Germany

Supporting Information

ABSTRACT: Molecular patterns are widely used for compound filtering in molecular design endeavors. They describe structural properties that are connected with unwanted physical or chemical properties like reactivity or toxicity. With filter sets comprising hundreds of structural filters, an analytic approach to compare those patterns is needed. Here we present a novel approach to solve the generic pattern comparison problem. We introduce chemically inspired fingerprints for pattern nodes and edges to derive an easy-to-compare pattern representation. On two annotated pattern graphs we apply a maximum common subgraph algorithm enabling the calculation of pattern inclusion and similarity. The resulting algorithm can be used in many different ways. We can automatically derive pattern hierarchies or search in large pattern collections for more general or more specific patterns. To the best of our knowledge, the presented algorithm is the first of its kind enabling these types of chemical pattern analytics. Our new tool named SMARTScompare is an implementation of the approach for the SMARTS language, which is the quasi-standard for structural filters. We demonstrate the capabilities of SMARTScompare on a large collection of SMARTS patterns from real applications.



INTRODUCTION

Chemical patterns are widely used during filtering steps of drug design endeavors. They are associated with the common task of substructure searching for filtering and tagging. There are a few published chemical pattern languages, such as SMARTS,[1] SLN,[2] and Molecular Query Language.[3] All of those languages have in common that they describe molecular substructures with many features for atoms and bonds. Several collections comprising hundreds of patterns have been published for different languages.[4−12] The primary focus of these collections is compound identification in a high-throughput manner.[5,8−12]

When using chemical patterns to filter molecules, the focus lies more on matching the right class of chemistry than identifying the specific atoms that are matched by pattern nodes. Patterns are furthermore frequently used to match functional groups[13−15] and identify and classify rotatable bonds[16,17] or single atoms with specific chemical environments.[7] In this context, the set of atoms that are matched by the pattern are of specific interest. Their environment can be important for the pattern match, although it is not part of the expected result.

For the SMARTS language, a series of tools have been developed for visualization[18,19] and editing.[20] Furthermore, some existing methods are able to automatically construct molecular patterns separating two compound sets,[21,22] the latter automatically creating SMARTS expressions. However, beyond visual inspection, there is no validated algorithmic approach available to date to compare two SMARTS patterns. To improve, compare, or extend pattern collections in a meaningful way, a pattern comparison method is highly desirable.

From a purely semantic point of view, molecular patterns are similar to regular expressions. Regular expressions are used extensively for text search in computer science. In theoretical computer science, inclusion and equality of regular expressions are computed, transforming them into equivalent deterministic final automatons (DFAs) and testing whether their accepted languages are inclusive or identical. However, there might be an exponential blowup while constructing the DFA.[23] For subsets of regular languages, there are polynomial time DFA constructions[24] and polynomial time comparison approaches for direct regular expression comparisons.[25] However, the structural differences between regular expressions and chemical patterns are too large to transfer these findings.

In cheminformatics, the RDKit[26] library has some functionality implemented to compare two molecular patterns in the form of SMARTS. Subset relations of simple patterns that are written as pure conjunctions without any negation can be determined if no property transfer is needed. For the more general pattern comparison problem, allowing disjunctions or negation of simple properties, the results do not guarantee to find more specific patterns anymore (for more details, see section SI6 in the Supporting Information).

Received: March 22, 2019
Published: May 23, 2019

DOI: 10.1021/acs.jcim.9b00250
J. Chem. Inf. Model. 2019, 59, 2560−2571

To be able to conduct comprehensive analysis of molecular patterns, we have developed a novel analytic approach. In the following, we will model the chemical pattern comparison problem and present an outline of our algorithm. Several technical details of the method can be found in the Supporting Information. Finally, we validate the detection of more specific patterns and conclude with an outline of upcoming applications for the SMARTScompare approach.[27]

METHODS

The Chemical Pattern Comparison Problem. In this section, an abstract chemical pattern is defined. This follows the goal to provide definitions for a generic pattern comparison algorithm that is applicable to several pattern languages. On the basis of the concept of abstract chemical patterns, we can define relevant terms like pattern isomorphism, pattern inclusion, and pattern similarity. The following definitions focus on the chemical space of existing molecules (CS) and do not consider differences introduced by different implementations of pattern languages or chemistry models.

For simplicity, we decided to focus our theoretical approach on the problem of identifying molecule sets matched by a certain pattern. Describing the individual atoms matched in a molecule by a pattern will be considered later when the theory is mapped to the SMARTS language.

Isomorphic Patterns. A chemical pattern P, such as a SMARTS expression, can be understood as a subset $CS(P)$ of the space of all theoretically existing molecules, CS. Some of the molecules from CS match the pattern and therefore are part of the subset, while others do not and are not. The subset $CS(P)$ is infinite in size for many patterns, but nevertheless, it provides a first glimpse of the notion of pattern equality: we consider two patterns P_1 and P_2 to be equal if and only if $CS(P_1) = CS(P_2)$. It becomes immediately clear that if $CS(P_1) = CS(P_2)$, then P_1 and P_2 have to match the same set of atoms in each molecule of $CS(P_1)$.

Since two patterns can match different parts of a molecule, further relationships like mutual exclusion are not particularly useful if defined this way. To capture the chemical intuition of a term saying "two patterns have something or nothing in common", we have to take the mappings between the nodes of the patterns and the atoms within the matching molecules into account. Since a pattern node always matches a single atom in a molecule, we can associate node n with the subset $AS(n)$ of all matching atomtypes from a whole atomtype space, AS. As above, we can define the equality relationship for a pair of pattern nodes n_1 and n_2. It should be noted that in contrast to CS, the space AS is of finite size, such that the node relationships can be computed easily. Once equality between nodes is defined, many concepts from graph theory, most importantly isomorphism and subgraph isomorphism, can be transferred to chemical patterns. For completeness, we can consider bondtypes analogously. For a pattern edge e, $BS(e)$ is the subset of all matching bondtypes from the whole bondtype space, BS. Two pattern edges e_1 and e_2 are considered equal if and only if $BS(e_1) = BS(e_2)$.

In summary, we suggest that pattern isomorphism be defined as follows: Given two chemical patterns $P_1 = (N_1, E_1)$ and $P_2 = (N_2, E_2)$, where the N_i are the sets of nodes and the E_i are the sets of edges between nodes, P_1 and P_2 are isomorphic if and only if there exists a bijective node-mapping function f: $N_1 \rightarrow N_2$ such that

- $\forall\, n \in N_1$: $AS(n) = AS(f(n))$
- $\forall\, m, n \in N_1$: $\{m, n\} \in E_1 \Leftrightarrow \{f(m), f(n)\} \in E_2 \wedge BS(\{m, n\}) = BS(\{f(m), f(n)\})$

Furthermore, we need the following restrictions:

1. The atomtype space AS and bondtype space BS cannot contain any chemically infeasible or redundant states.
2. Some properties (e.g., aromaticity) have a strong influence on the environment of a node or an edge, e.g., if a pattern describes exactly two generic atoms that are part of a generic aromatic ring, then its second aromatic node does not provide any information to the pattern. Because of this, a generic chemical pattern must be nonredundant in several ways:
   (a) Each node of a chemical pattern must be relevant for its representation in the chemical space CS.
   (b) Chemical patterns must be nonredundant in the sense that for each node n, its whole subset of atomtype space, $AS(n)$, is necessary to describe $CS(P)$. The same must hold true for all of the pattern edges e and their subsets of the bondtype space, $BS(e)$. Otherwise different patterns P_1 and P_2 ($P_1 \neq P_2$) could describe the same subset of chemical space, i.e., $CS(P_1) = CS(P_2)$. This implies that for each atomtype and bondtype that is possible for any pattern node or edge, there exists a molecule $m \in CS(P)$ such that $m \notin CS(P')$ if a certain atomtype or bondtype is not present at the node or edge:

$$P = (N_1, E_1) \tag{1}$$

$$\forall\, n \in N_1\colon \forall\, s_a \in AS(n)\colon \exists\, m \in CS(P)\colon \exists\, P'\colon P' = (N_1 \backslash \{n\} \cup \{n \backslash \{s_a\}\}, E_1) \Rightarrow m \notin CS(P') \tag{2}$$

$$\forall\, \{m, n\} \in E_1\colon \forall\, s_e \in BS(\{m, n\})\colon \exists\, m \in CS(P)\colon \exists\, P'\colon P' = (N_1, E_1 \backslash \{\{m, n\}\} \cup \{\{m, n\} \backslash \{s_e\}\}) \Rightarrow m \notin CS(P') \tag{3}$$

The Subpattern Relationship. From the application point of view, it would be very interesting to detect subset relationships. In practice, a subset relationship $P_1 \subseteq P_2$ means that P_2 is a more generic pattern than P_1, that is, P_2 matches all molecules that P_1 hits and potentially some more (in which case P_1 is a true subset). For a pair of pattern nodes n_1 and n_2, we define $n_1 \subseteq n_2$ if and only if $AS(n_1) \subseteq AS(n_2)$. Following the definition of subgraph isomorphism, we are able to define the subset relationship between chemical patterns as follows: We consider a chemical pattern $P_1 = (N_1, E_1)$ to be a subpattern of $P_2 = (N_2, E_2)$ if and only if there exists a subset $N_1' \subseteq N_1$ and a bijective node-mapping function f: $N_2 \rightarrow N_1'$ such that

- $\forall\, n \in N_2$: $AS(f(n)) \subseteq AS(n)$
- $\forall\, n_1, n_2 \in N_2$: $\{n_1, n_2\} \in E_2 \Leftrightarrow \{f(n_1), f(n_2)\} \in E_1 \wedge BS(\{f(n_1), f(n_2)\}) \subseteq BS(\{n_1, n_2\})$

Every node and edge of the more generic pattern P_2 is mapped to a node of the more specific pattern P_1. Each single mapping follows the subset relation as well. It should be noted that the intuitive relationship between pattern isomorphism and

DOI: 10.1021/acs.jcim.9b00250
J. Chem. Inf. Model. 2019, 59, 2560−2571

subpattern isomorphism holds, i.e., if $P_1 \subseteq P_2$ and $P_2 \subseteq P_1$, it follows that $P_1 = P_2$.

The Maximum Common Subpattern. $P_{sub} = (N_{sub}, E_{sub})$ is the maximum common subpattern of pattern $P_1 = (N_1, E_1)$ and pattern $P_2 = (N_2, E_2)$ if and only if there exists a subset $N_1' \subseteq N_1$, a subset $N_2' \subseteq N_2$, and bijective node-mapping functions $f_1: N_{sub} \rightarrow N_1'$ and $f_2: N_{sub} \rightarrow N_2'$ such that

- $\forall\, n \in N_{sub}: AS(n) \subseteq AS(f_1(n))$
- $\forall\, n \in N_{sub}: AS(n) \subseteq AS(f_2(n))$
- $\forall\, n_1, n_2 \in N_{sub}: \{n_1, n_2\} \in E_{sub} \Leftrightarrow \{f_1(n_1), f_1(n_2)\} \in E_1 \wedge BS(\{n_1, n_2\}) \subseteq BS(\{f_1(n_1), f_1(n_2)\})$
- $\forall\, n_1, n_2 \in N_{sub}: \{n_1, n_2\} \in E_{sub} \Leftrightarrow \{f_2(n_1), f_2(n_2)\} \in E_2 \wedge BS(\{n_1, n_2\}) \subseteq BS(\{f_2(n_1), f_2(n_2)\})$
- N_{sub} and E_{sub} cannot be extended.

Pattern Similarity. To capture the chemical intuition of similarity, the definition of pattern similarity needs to include more aspects than pattern representations in *CS*. For example, two patterns might match only different molecules that are, however, very similar to each other. In other words, chemical similarity should be reflected in pattern similarity although the patterns match different molecules.

To mimic the chemical intuition of similar patterns, we propose a probability model of extended atomtype occurrence. We therefore define similarity between patterns P_1 and P_2 such that the score approximates the coverage of P_1 in P_2 using the maximum common subpattern P_{sub}. On the basis of a statistic of the occurrences of extended atomtypes, context-based similarity of nodes can be defined. We propose a similarity measure that approximates weighted coverage of $AS(n_{sub})$ in $AS(n_1)$ for compatible nodes of two chemical patterns: two nodes n_1 and n_2 are similar if and only if $AS(n_1) \cap AS(n_2) \neq \emptyset$. This definition supports the intuitive relationship $CS(P_1) = CS(P_2) \Rightarrow \mathrm{Sim}(P_1, P_2) = 1$.

COMPARING SMARTS EXPRESSIONS

Over the past decades, the SMARTS language has been established as the quasi-standard for molecular patterns. Concepts like logical expressions for atoms and bonds as well as recursive pattern definition make SMARTS an extremely powerful language. The SMARTScompare algorithm presented here will cover most of the language constructs and solve the various pattern comparison problems defined above.

SMARTScompare is based on the calculation of a maximum common substructure (MCS) between the two pattern graphs. In the following, we will use the terms *node* and *edge* whenever we mean the corresponding components of a SMARTS expression, while *atom* and *bond* are used for molecules only.

The overall strategy of SMARTScompare can be summarized as follows: First, a method is required that allows two single SMARTS nodes to be compared with each other, and here a fingerprint representing the SMARTS atomtype space *AS* is developed for this purpose. After this concept is extended to SMARTS edges, an MCS algorithm is applied. By modification of the node compatibility function, all of the problems specified above, including the similarity calculation, can be addressed with a single algorithm. Finally, we extend the algorithm to deal with recursive SMARTS expressions.

Fingerprint Generation and Pattern Preparation. *The Extended Atomtype.* The comparison of SMARTS expressions is based on a description of chemical space following a standard chemistry model.[28] Atoms receive an atomtype with a corresponding valence state covering the element, the number and types of incident bonds, and the charge. The SMARTS language is able to distinguish atoms in more detail, so the atomtype is extended with the following SMARTS-specific properties: number of attached hydrogens (no distinction between explicit and implicit hydrogens), number of aromatic bonds, hybridization, number of rings of which the atom is a part, minimum ring size, and number of ring bonds. While the original SMARTS ring features are based on the smallest set of smallest rings (SSSR)[29] concept, which is known to be ambiguous, we employ the unique ring families (URF) approach.[30] Hybridization is not part of the original SMARTS definition, but it is supported by many implementations and therefore is included in the extended atomtype states. The extended atomtypes additionally cover the number of incident aromatic bonds. This is not part of the SMARTS language but improves the distinction of atomtypes in aromatic ring systems. Although technically possible, our current implementation does not support metals, isotopes, or chirality. Figure 1 gives an example of extended atomtypes uniquely assigned to atoms of a molecule. All of the chemically feasible extended atomtypes are enumerated and stored in a list. Table 1 lists the applied criteria for chemical feasibility. Properties like connectivity ("X") that are covered by the chemistry model's valence state or atomtype need no further enumeration and are directly used in the extended atomtypes. Properties like adjacent hydrogen ("H" and "h") that can be limited on the basis of other properties of the atomtype are enumerated in this explicit range. Finally, for the ring properties ("R" and "r") there is no such limitation.

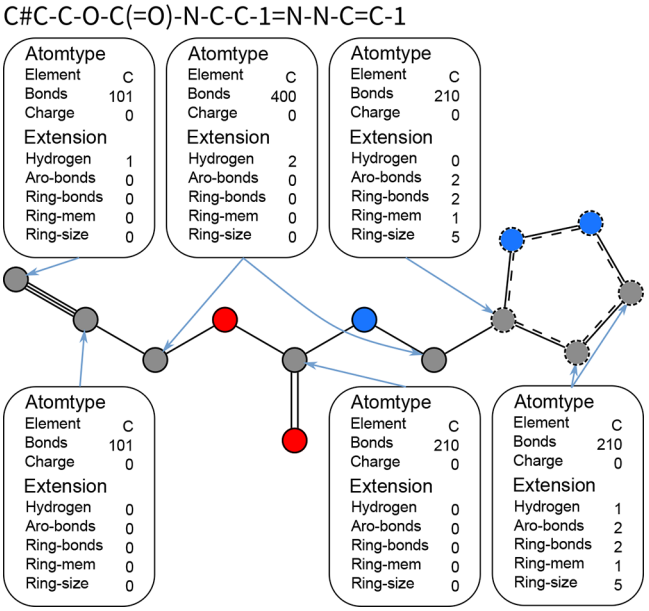


Figure 1. Example of a molecule with the unique annotations of extended atomtypes at the carbon atoms. The three-digit bond code is the number of localized single, double, and triple bonds.

Table 1. Criteria for Chemical Feasibility of the Enumerated Extended Atomtypes

$n_{Hydrogen} + n_{Aro\text{-}bonds}$	$\leq$	n_{Bonds}
$n_{Hydrogen} + n_{Ring\text{-}bonds}$	$\leq$	n_{Bonds}
$n_{Aro\text{-}bonds}$	$\leq$	$n_{Ring\text{-}bonds}$
$n_{Ring\text{-}bonds}/2$	$\leq$	$n_{Ring\text{-}mem}$

DOI: 10.1021/acs.jcim.9b00250
J. Chem. Inf. Model. 2019, 59, 2560−2571

Their enumeration is artificially limited to at most four URFs (rings) of which an atom is a part and a maximum ring size of 24 atoms. There are additional extended atomtypes for atoms in more than four URFs and in rings with a minimum size larger than 24, but they can no longer be distinguished from each other. Table 2 presents all of the covered SMARTS node properties and how they are represented in the extended atomtypes. The enumeration of all feasible feature combinations results in 20 565 unique extended atomtypes, called the atomtype space. A more detailed enumeration scheme is given in the Supporting Information (see SI1).

Table 2. SMARTS Node Properties and How They Are Represented in the Extended Atomtypes

SMARTS node property	symbol	represented in extended atom type
Aromaticity	a, ...	enumerated property
Charge	+, −	part of the atomtype (chemistry model)
Connectivity	X	sum over incident bond types (chemistry model)
Degree	D	difference of connectivity and hydrogen property
(ExplicitlImplicit) hydrogen	H, h	enumerated property
Hybridization	^	enumerated property
Ring connectivity	x	enumerated property
Ring size	r	enumerated property
Ring membership	R	enumerated property
Atomic number	#	part of the atomtype (chemistry model)
Valence	v	weighted sum over incident bond types (chemistry model)

The atomtype space is the basis of a fingerprint descriptor for SMARTS nodes consisting of 20 565 bits, in which each bit corresponds to a specific extended atomtype as depicted in Figure 2. This descriptor captures any logical expressions the SMARTS node description might contain.

Basic Fingerprint Generation. The generation of the specific SMARTS node fingerprints consists of two major parts (see Figure 3):

1. All bits corresponding to extended atomtypes fulfilling the SMARTS node expression are set to active.
2. All set bits violating properties of the environment of the SMARTS node are reset.

Table 3 lists some criteria used for bit reduction in the second step. They are also part of an extended list given in Table S4 in the Supporting Information. How these properties are retrieved from the node environments and postprocessed is also shown in the Supporting Information (see SI2).

For each edge of a SMARTS expression, a fingerprint of nine bits representing bond features is created. Edge fingerprint bits represent single, double, triple, and directed bonds in and outside of rings and aromatic ring bonds. The first four bits belong to nonring bonds and the following four bits to ring bonds in the given order. The ninth bit represents an aromatic ring bond. The edge fingerprint is depicted in Figure 4.

The resulting edge fingerprint can be further pruned by examining the neighboring nodes and edges. If any of the incident nodes does not allow a certain bondtype in its environment, the corresponding bits in the edge fingerprint are reset. After that, incident nodes and edges that could be affected by an update are updated as well until convergence is reached. The node fingerprint update is based on the environment criteria listed in Tables 3 and S4. The criteria for the edge fingerprint updates are listed in Table 4.

We want to highlight that fingerprint pruning is necessary for subset detection. The pruning strategy does not guarantee that each bit in all fingerprints is necessary to describe $CS(P)$ as defined for the generic chemical pattern in eqs 2 and 3. It should be noted that the pruning strategy is conservative since all pruned atom- or bondtypes are irrelevant to describe $CS(P)$.

Small rings up to a size of eight nodes can be detected as aromatic rings. If every node of a ring is aromatic, the edge fingerprints are pruned to reset all bits representing non-aromaticity. This feature allows SMARTS expressions like c1ccccc1 and c:1:c:c:c:c:c:1 to be considered as equal. In general, there is no possibility to detect a generic edge as aromatic within the SMARTS context. In macrocycles there can be aliphatic single bonds between aromatic atoms. Consequently, it is necessary to restrict the supported ring sizes for this feature. According to the NAOMI chemistry model,[28] the limit is set to a maximum number of eight nodes for those rings. This feature is enabled by default in SMARTScompare because it is cumbersome to mark aromaticity explicitly and most often it is

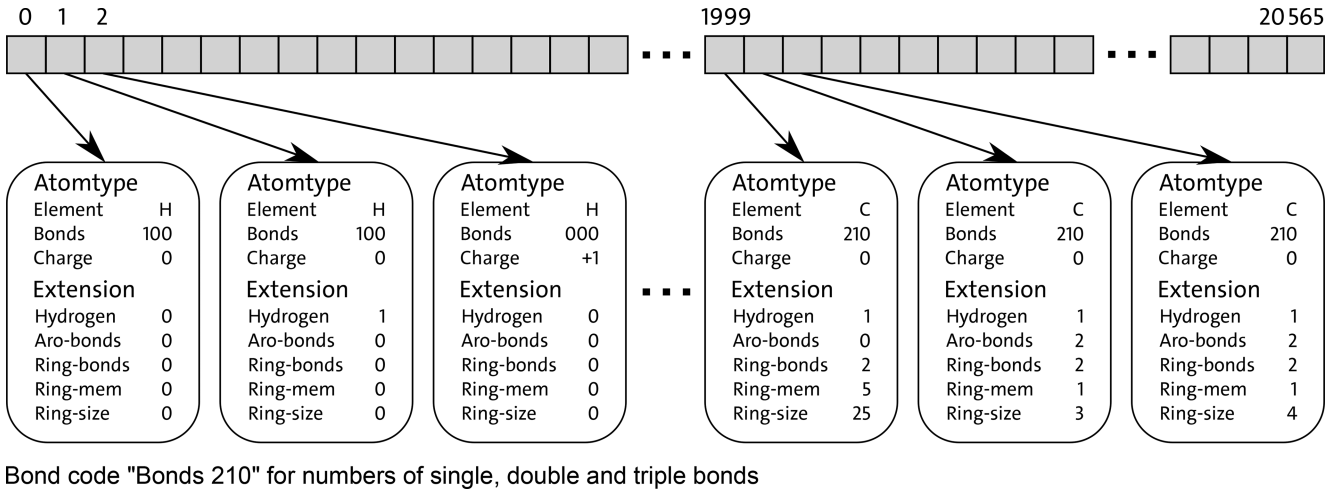


Figure 2. Depiction of the systematically enumerated 20 565 extended atomtypes. The assigned indices correspond to the implementation list position. Hybridization is omitted in the depiction.

DOI: 10.1021/acs.jcim.9b00250
J. Chem. Inf. Model. 2019, 59, 2560−2571

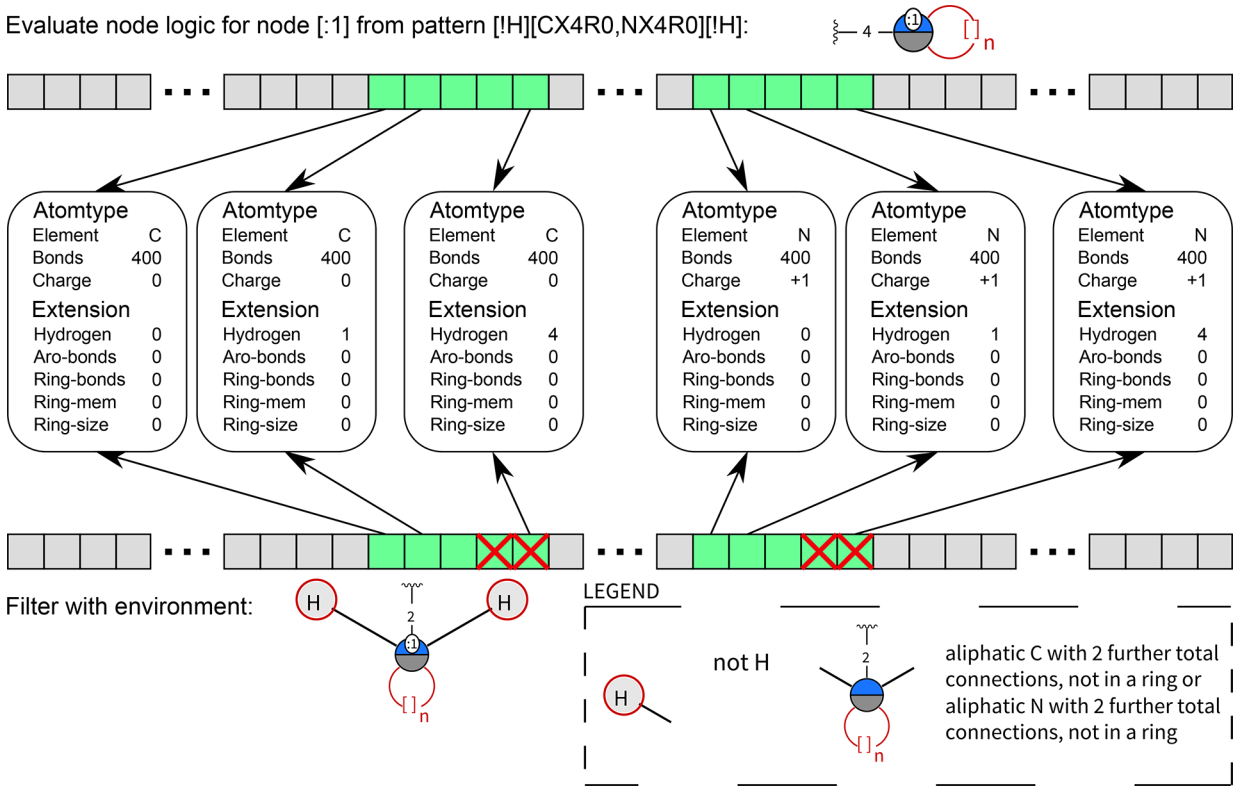


Figure 3. Depiction of the fingerprint generation for the second node (shown at the top) of the pattern [!H][CX4R0,NX4R0][!H]. Below the pattern, the fingerprint is shown. In the center of the figure there are some extended atomtypes corresponding to the bits set in the fingerprint. The lower depiction of the fingerprint shows reset bits of the fingerprint belonging to atomtypes that are not compatible to the environment of the node (see the pattern depiction at the bottom left). Pictures were created with SMARTSviewer.[18,19]

Table 3. Criteria for Fingerprint Bits That Arise from the Node Environment

node environment		extended atomtype
env_{Bonds}	$\leq$	n_{Bonds}
env_{Valence}	$\leq$	$n_{n_{\text{Bonds}}\text{alence}}$
env_{Hydrogen}	$\leq$	n_{Hydrogen}
$env_{\text{!Hydrogen}}$	$\leq$	$n_{\text{!Hydrogen}}$
$env_{\text{Double-bond}}$	$\leq$	$n_{\text{Double-bond}}$
$env_{\text{Triple-bond}}$	$\leq$	$n_{\text{Triple-bond}}$
$env_{\text{Valence}} - env_{\text{Degree}}$	$\leq$	$n_{\text{Valence}} - n_{\text{Degree}}$
$env_{\text{Aro-bond}}$	$\leq$	$n_{\text{Aro-bond}}$
$env_{\text{Ring-bond}}$	$\leq$	$n_{\text{Ring-bond}}$
$env_{\text{!Ring-bond}}$	$\leq$	$n_{\text{!Ring-bond}}$

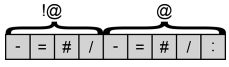


Figure 4. Edge fingerprint. The first four bits represent acyclic bondtypes, and the final five bits represent the cyclic ones. The bits are annotated with their meaning in SMARTS symbols.

Table 4. Edge Fingerprint Environment Pruning Criteria[a]

edge type	criterion
Single bond	both incident nodes allow single bonds
Double bond	both incident nodes allow double bonds
Triple bond	both incident nodes allow triple bonds
Directed bond	any incident node allows double bonds
Aromatic bond	both incident nodes can be aromatic
Ring bond	both incident nodes can be in rings

[a]If any incident node of an edge does not allow a certain bondtype in its environment, the corresponding bits in the edge fingerprint are reset. For example, if the fingerprint of an node incident to an edge contains no extended atomtype with incident single bonds, the edge cannot be a single bond.

implicitly part of SMARTS patterns. As shown in Validation, this feature can sometimes lead to erroneous subset claims.

SMARTS recursion has an influence on fingerprint generation too. The effects are described briefly after the matching algorithm and in detail in the Supporting Information (see SI3).

The Matching Algorithm. The mapping between two SMARTS patterns is calculated using a clique-based connected maximum common induced subgraph approach.[31−33] In the first step, a product graph, often called a compatibility graph, is built. Cliques in this graph represent common subgraphs between the two SMARTS expressions. For graphs, this is a relatively simple step since only the labels of nodes and edges are compared. For SMARTScompare, the compatibility between nodes and edges depends on the problem to be solved. In all cases, we use the subset of all matching atomtypes $AS(n)$ for a node n or bondtypes $BS(e)$ for an edge e. The compatibility criteria are listed in Table 5

For pattern isomorphism, we expect the MCS to be complete, i.e., to match all nodes of the first pattern to all nodes of the second. For subpattern isomorphism, all nodes of the first pattern (the more generic one) have to be matched. In all other cases, we search for the maximum connected common subgraph. By the use of a modified connected MCS algorithm of the Parasols library,[34] all MCSs of maximum size are returned as the result.

DOI: 10.1021/acs.jcim.9b00250
J. Chem. Inf. Model. 2019, 59, 2560−2571

Table 5. Fingerprint Compatibility Criteria for the Different Problem Types

	compatibility criteria	
comparison problem	for two nodes n_1 and n_2	for two edges e_1 and e_2
pattern isomorphism	$AS(n_1) = AS(n_2)$	$BS(e_1) = BS(e_2)$
subpattern isomorphism	$AS(n_2) \subseteq AS(n_1)$	$BS(e_2) \subseteq BS(e_1)$
similarity search	$AS(n_1) \cap AS(n_2) \neq \emptyset$	$BS(e_1) \cap BS(e_2) \neq \emptyset$

An example execution of the algorithm performing a similarity search between two patterns is summarized in Figure 5. Figure 6

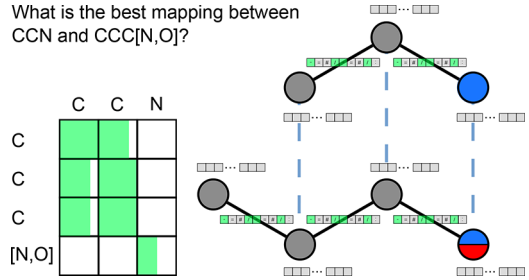


Figure 5. Matching procedure for nonrecursive SMARTS patterns. For each node and edge, a fingerprint is generated, and the compatibility graph is built. In this figure it is represented by the compatibility matrix on the left. In a similarity search, two nodes are compatible if their fingerprints share at least one bit. Partially filled cells of the matrix indicate the percentage of shared bits. Finally, the MCS algorithm computes all cliques of maximum size, thus computing the maximum common subgraph of the two SMARTS patterns.

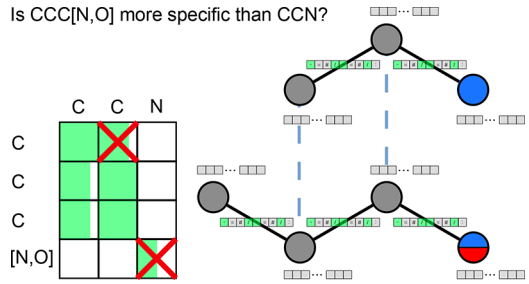


Figure 6. Same matching procedure as in Figure 5, but for the subset search. This employs additional constraints, such that each node of CCC[N,O] must be more specific than compatible nodes of CCN. The two nodes that are not more specific are crossed out. Beyond that, the computation stays the same. CCC[N,O] is not a subset of CCN, since the mapping does not completely cover CCN.

shows a similar example executing a subset search for more specific patterns. These two examples emphasize the algorithmic similarity between the available execution modes similarity search and subset search: there is no difference except for the compatibility criterion. In both figures, the symbolic annotation of fingerprints is performed first. Then the compatibility graph, represented as a compatibility matrix, is created. Finally, the maximum common subgraph is depicted (blue connection lines).

Result Compilation. The algorithm computes all maximum common subgraphs for each comparison with the patterns and their recursions. On the basis of these MCS results, the final mapping result is produced. The computed mapping between two patterns should be independent of pattern-invariant aspects of the SMARTS string, such as node order, order of logic operands, etc. This is achieved by enumerating all possible mappings and scoring them. The fingerprint pairs of all mapped nodes are scored using a similarity function like the Tanimoto coefficient. The result yielding the highest score is finally chosen.

In the case of subset search including recursive patterns, additional constraints have to be applied. They are described in the Supporting Information (see SI5).

Finally, there are a few more aspects that are handled during result compilation. The definition of the generic chemical pattern contains several constraints regarding pattern uniqueness. By pruning the SMARTS node and edge fingerprints, SMARTScompare tries to come as close as possible to a generic pattern representation. This is not always achievable. Explicit hydrogen nodes ([H] or [#1]) are always included in their neighbor's node fingerprints. Thus, they can be considered redundant. The same holds true for wildcard nodes connected with exactly one acyclic edge. Since their fingerprints are completely determined by their environment, they can be considered redundant as well.

SMARTS Recursion. SMARTS recursion is a node property that describes the structural environment of the node to which it is attached. The recursive description, a SMARTS pattern by itself, is compared against the molecule, keeping the first node assigned to the atom matched while matching the recursive description independent from the main pattern (for a detailed description of SMARTS recursion, see SI3−SI5).

Recursive environment descriptions are frequently used, but unfortunately, they substantially complicate the comparison of SMARTS patterns. To handle SMARTS recursion correctly, the matching procedures are usually recursive by themselves. To integrate SMARTS recursion into the comparison algorithm, fingerprints influenced by SMARTS recursion have to be modified, the MCS algorithm has to be implemented in a recursive fashion, and the logic between recursive expressions has to be taken into account.

Recursion Fingerprint Handling. Node environments formulated in SMARTS recursion have an influence on fingerprint generation. Positive environments participate with their first node, since it has to be fulfilled at the attachment node. Negated environments have to be considered for fingerprint generation only if they represent exactly one node; otherwise, they do not influence any node fingerprints.

SMARTS recursion is explicitly considered after the first fingerprint pruning step for the basic pattern. Handling of SMARTS recursion is divided into the following steps:

1. Generation of node and edge fingerprints for the recursive environment pattern.
2. Removal of redundant SMARTS recursion, i.e., recursive environments that are more generic than the basic pattern.
3. Evaluation of the logic in the node expression and restructuring into a disjunctive normal form (DNF). In this context, alternatives are defined as conjunctions of SMARTS recursion, and the node expression is a disjunction of alternatives.
4. Removal of redundant SMARTS recursions within the formulated alternatives.
5. Removal of redundant alternatives from the generated DNFs in SMARTS node expressions.
6. Update of node fingerprints using all of the information on SMARTS recursion grouped into alternatives.
7. Final pruning of all fingerprints using the updated fingerprints of nodes with SMARTS recursion.

DOI: 10.1021/acs.jcim.9b00250
J. Chem. Inf. Model. 2019, 59, 2560−2571

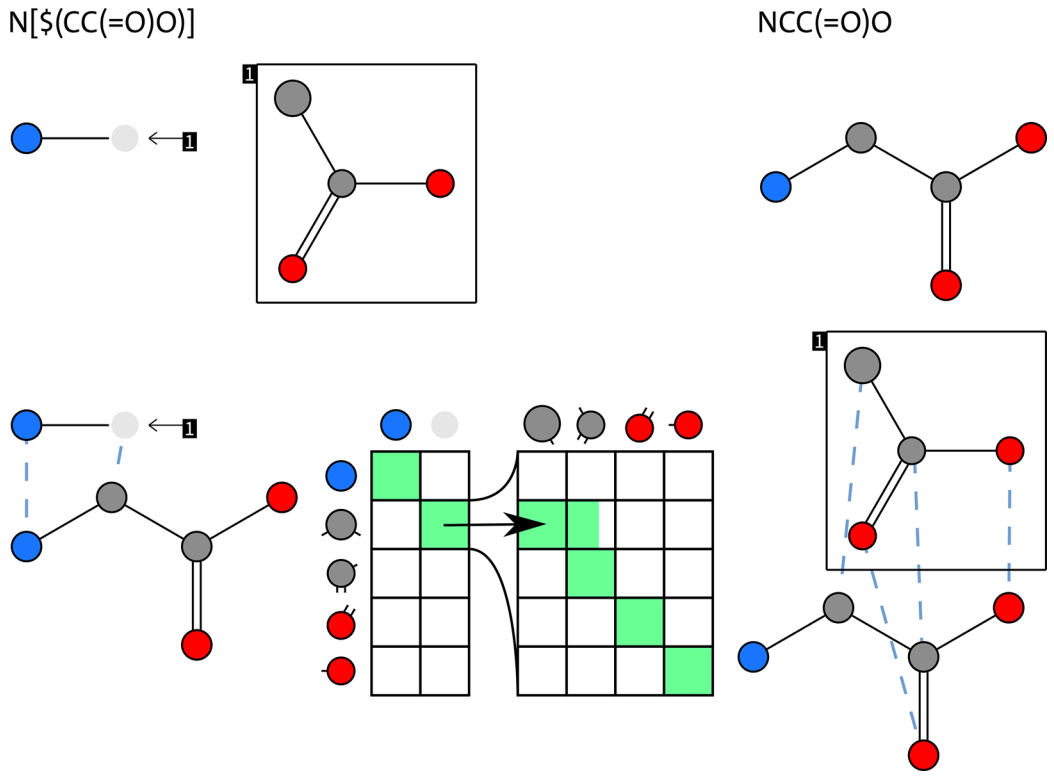


Figure 7. Recursive matching procedure for the comparison of N[$(CC(=O)O)] and NCC(=O)O. The patterns are shown at the top. At the bottom left, the MCS of the structures is shown. In the bottom middle, the compatibility matrix for the structure compatibility and the nested compatibility matrix for the MCS calculation of the recursion of the first pattern and the structure of the second one are shown. At the bottom right, the MCS of the recursion and the structure is depicted. Whenever a recursive node's fingerprint is compatible with another node, the compatibility is finally derived by performing the matching recursively. This means that nodes are compatible if and only if their environments are compatible.

A detailed description of the SMARTS recursion handling during atomtype fingerprint generation is given in the Supporting Information (see SI3).

Recursive Matching. To determine the compatibility of nodes with recursive environments, the matching algorithm is recursive by itself. If at least one of the mapped nodes carries a SMARTS recursion, the compatibility of the nodes is determined by comparing all required combinations of the SMARTS recursive environments. A general overview of this recursive matching can be seen in Figure 7. The first pattern has two nodes, one of them with SMARTS recursion, and the second pattern does not have a recursive expression. The compatibility of the node with SMARTS recursion and a fingerprint-compatible node is a result of a recursive comparison of the SMARTS recursion of the first pattern and the second pattern. This is shown by the two compatibility matrices in Figure 7.

To handle SMARTS recursion correctly, the procedure has to solve the following tasks:

1. Matching a SMARTS recursion to the basic pattern.
2. Matching a SMARTS recursion to another SMARTS recursion.
3. Evaluating the pattern logic to derive constraints for the subset search.
4. Correcting results with respect to negated SMARTS recursions.

A detailed description of all mentioned procedures is given in the Supporting Information (see SI4).

Finally, nodes are compatible if their fingerprints are compatible and their SMARTS recursions are compatible with respect to the comparison mode of interest (subset or similarity).

The presented procedure follows an indirect recursion that occurs when the compatibility of nodes with SMARTS recursion is determined. The MCS algorithm needs to determine the node compatibility of all nodes, including those with SMARTS recursion. To determine the compatibility of nodes with SMARTS recursion, the SMARTScompare algorithm is applied to the recursive SMARTS pattern.

SMARTS Recursion during Result Compilation. The matching procedure results in several partial results for the basic pattern and the recursions. For the final result, an optimal mapping between these patterns is required. A detailed description how this mapping is computed is given in the Supporting Information (see SI5).

Further Features. The algorithm described above was designed for SMARTS comparisons. However, during its development we noticed its potential for error detection in SMARTS expressions:

- The exhaustive atomtype state generation is a powerful chemical feasibility verification for SMARTS expressions. This procedure verifies that each node corresponds to at least one chemical state. This test includes the node's recursive environment as well as the node's logical expression. In general, SMARTS node expressions exceeding element-specific valences and impossible logical constructions are detected. For example, the algorithm will detect the nitrogen in patterns like C[$(N(=O)=O)] as a nitrogen with a valence of 5, which is unsupported by the chemistry model.

DOI: 10.1021/acs.jcim.9b00250
J. Chem. Inf. Model. 2019, 59, 2560−2571

- The algorithm is able to detect many types of redundancy in SMARTS expressions. On the basis of this information, SMARTS expressions can be optimized with respect to computational complexity and human readability. This includes even some kinds of redundancy within the logical constructions of SMARTS recursion. If a recursion is more generic than the environment of its attachment node, it does not specify the pattern and thus could be omitted, as in [$(CO)]O. If SMARTS recursions in logical structures are more specific or more generic, they could be redundant as well. For example, the pattern [(CO)(C*),$(C*)] gives the same description as [$(C*)]. More details are provided in the Supporting Information (see SI3).

GENERALITY OF THE APPROACH

The above-described algorithm for SMARTS pattern comparison works correctly in most practical applications, but it is not complete and correct in general. In this section, we summarize unsupported features and cases where the algorithm fails.

As mentioned before, all kinds of chirality and isotopes are unsupported to date. Isotopes are easy to include in the fingerprint approach, but they are unsupported within the chemistry model of the underlying NAOMI[28] cheminformatics library. Disconnected SMARTS patterns are also unsupported by the algorithm. Chirality is not considered during fingerprint generation. The verification of ligand orders, which is required for chirality handling, is currently unsupported. In order to extend the algorithm in this direction, it would be necessary to verify the ligand orders within the MCS algorithm or in the final result compilation. Currently our implementation denies subset relations of patterns including those unsupported SMARTS features.

One core aspect of the SMARTScompare algorithm is extensive valence counting. With regard to unspecific edges, the abilities to detect pattern identity are limited. As shown in Table 4, edge fingerprints are pruned with regard to their neighboring nodes. This kind of pruning is not chemically aware of whether a certain bondtype is allowed for a specific edge. It is only aware of the existence of a specific bondtype within the set of incident edges of a node.

Saturation of a bondtype is undetected. This might lead to missed pattern identities, as illustrated in the comparison of the patterns *=[CX3]~[CX3] and *=[CX3]-[CX3]. In this example, the algorithm does not detect that the wildcard bond has to be a single bond resulting in pattern identity. A similar example is shown in Figure 8. The two oxygen atoms can only be connected via a double bond and a single bond. To solve this issue, it is necessary to detect and break the symmetry. Both examples can be extended to a general case where SMARTScompare is unable to detect pattern identities because some fingerprints still require pruning to fulfill the non-redundancy requirements described in eqs 2 and 3.

A generic example of patterns of this class exploits the fact that a certain bond occurs exactly once at all incident bonds to an atom. Then there have to be at least two incident edges if those fingerprints allow two different bondtypes. If the pattern is symmetric, then SMARTScompare is unable to detect pattern identity with a localized variant. We can compare the patterns O~[CX3v4H1]~O and O=[CX3v4H1]O, which are not detected as equal by SMARTScompare. Since the pattern with the wildcard edges is symmetric and there is exactly one double bond incident to the central carbon node, the two patterns are actually identical.

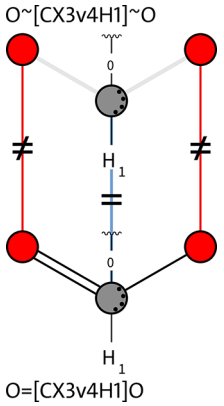


Figure 8. Two SMARTS patterns for formic acid that are not detected as equal. The upper pattern is written more generically than necessary. The pattern is symmetric, and the two wildcard bonds can only be a single bond and a double bond. Thus, neither the edges nor the two nodes describing oxygen are detected as equal.

Compatibility of logical constructions within environments can be hard to determine, and the approach might miss subset relations. There are constructions that are known to be undetectable, such as [$(C[N,O])] and [$(CN),$(CO)]. These two SMARTS match if there is a CN or a CO substructure. The subset relation is detected from the pattern with one recursion to the pattern with two recursions but is missed for the other way around because of algorithmic restrictions for node compatibility. Figure 9 depicts the situation. Our algorithm compares recursive expressions individually but is not able to combine chemical information on two recursive expressions into a single one. In general, if there are two alternatives for recursive SMARTS expressions that differ in only one node, the algorithm is unable to detect pattern identity to a pattern where the two recursions are combined into one recursion with the alternative combined within the node.

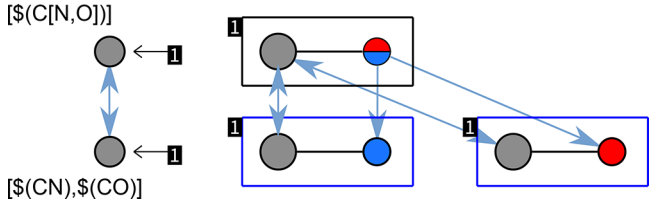


Figure 9. One limitation of the approach is that the subset search is strict and enforces the subset relation for every node. Although the two patterns shown are equal, the approach detects the subset relation in only one direction.

Another kind of undetected subset relation is shown in Figure 10. Besides the recursion, the two patterns are identical. Thus, all nodes of the structure should be compatible. However, the two nodes of the lower pattern connected by the cyan arrows have different fingerprints, although they can always match the same atom. In fact, the fingerprint of the node in the recursion ('[NH1]') is more general than its counterpart in the basic pattern. However, chemical information from neighboring nodes of the attachment node is not used to prune fingerprints in recursions and the other way around from neighboring nodes of recursions' root nodes. Because of this fingerprint pruning limitation, c−!@[NH1]C(=O)N is not detected to be equal to

DOI: 10.1021/acs.jcim.9b00250
J. Chem. Inf. Model. 2019, 59, 2560−2571

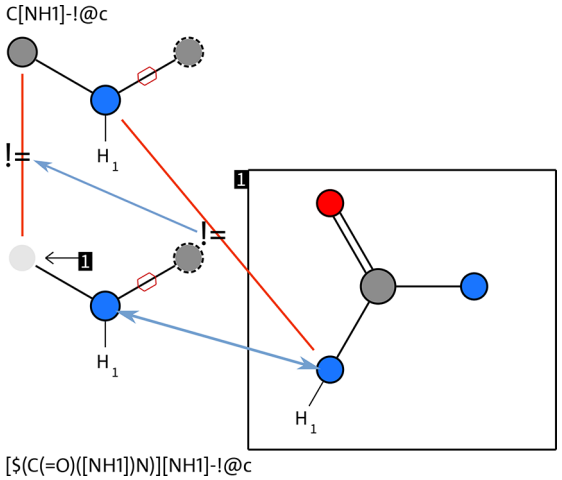


Figure 10. Another limitation of subset detection. For a recursive pattern which is compared with its nonrecursive structure, one of the recursions nitrogen nodes ('[NH1]') should have the same fingerprint as the corresponding node of the pattern's structure. Currently SMARTScompare does not compute recursion to structure mappings within a pattern to improve fingerprint pruning. This leads to a situation where the recursion is not mappable to the other pattern, and thus, the subset relation is missed. For this kind of error, it is necessary for nodes of the recursion to be described the same as nodes of the nonrecursive pattern without being redundant.

[$(C(=O)([NH1])N)][NH1]-!@c. Another example of a missed pruning opportunity in the basic pattern is the fact that [NH2][CH2]N is not detected to be equal to N[$([CH2]-[NH2])]N. This limitation holds for generic recursion expressions that do map parts of the environment of their attachment node. Their fingerprints could be pruned with the corresponding adjacent nodes of the attachment node only if the mapping is unique. The other way around with a unique mapping, fingerprints in the basic pattern could be pruned as well.

In general, negated SMARTS recursions have no influence on fingerprints of neighboring nodes or incident edges. Consider the pattern O[C!$(C=C)](N)~C. The wildcard bond C~C could be pruned with the knowledge of the negated SMARTS recursion !$(C=C) and become C-C. This kind of pruning using negated SMARTS recursion is not supported.

Negated SMARTS recursion is supported by the algorithm, but it is only partially considered in similarity searching. Similarity values do not consider negated SMARTS recursion except when there is a clear incompatibility. For subset search, negated SMARTS recursion in the more generic pattern and in the more specific pattern is supported.

The implementation of negated SMARTS recursion handling is designed to avoid deniable subset relations. Thus, negated recursion might lead to undetected subset relations. There are three criteria that allow subset detection for generic patterns with negated SMARTS recursion:

1. There is a compatible negated recursion in the more specific pattern.
2. The alternative of which the negated SMARTS recursion is part is not needed to detect compatibility.
3. The negated SMARTS recursion root node is incompatible with the mapping partner of its attachment node.

For any negated SMARTS recursion that violates all three criteria, subset relations are not detected. These criteria might be too strict to detect all possible subset relations, but they ensure that there are no false positives. Except for the pruning possibility, negated SMARTS recursion in the more specific pattern has no influence on the subset detection since it always specifies a pattern.

VALIDATION

Since no comparable algorithmic approach is available, we decided to approximately validate our approach using a large compound collection. On the basis of a large data set, one can determine candidates for subpattern relationships. If two patterns P_1 and P_2 both match several compounds of a diverse compound set and each molecule matched by P_2 is also matched by P_1, then P_2 is a candidate to be more specific than P_1. If there is any molecule matched by P_2 that is not matched by P_1, then P_2 is not more specific than P_1.

Validation of Subset Relationships. The compound set used contained ~370 million molecules and comprised all of the ZINC15[35] 2D compounds that were available on June 8, 2017. This set is called the *validation data set*. The collection of patterns was based on the data used for a systematic benchmark of subgraph algorithms[36] and contained 80 073 patterns in total. It consisted of SMARTS patterns as well as substructure-like patterns without any SMARTS properties, and 38 941 of those patterns matched at least one molecule in the validation data set. Patterns matching exactly one hydrogen were discarded and not part of that group of patterns. Of the matching patterns, 3393 used explicit SMARTS properties and the remaining ones were substructures without SMARTS properties.

This experimental setup is used to provide data for possible subset relations between SMARTS patterns. We sorted the entries for patterns by the number of matching molecules in descending order and extracted pattern pairs with potential subset relationships. There are 3 432 874 pattern pairs for which the sets of matched molecules are in a subset relationship. SMARTScompare determined 1 830 553 pattern pairs. The confusion matrix is shown in Table 6. In majority of the comparisons, the experiment and SMARTScompare agree that patterns are not in a subset relation. This is as expected using a diverse pattern set. For 53.3% of the experimental subset pairs, SMARTScompare also computes this relation. There are seven pattern pairs where the experiment proved SMARTScompare to be wrong (see Table 7). Six of them have the pattern [S;D3](-

Table 6. Comparison of the Subset Relations Detected by SMARTScompare and the Reference Data[a]

		experiment		
		S	N	total
SMARTScompare	S	1 830 546	7	1 830 553
	N	1 602 328	1 512 968 600	1 514 570 928
	total	3 432 874	1 512 968 607	1 516 401 481

[a]"S" stands for "subset relation" and "N" for "no subset relation".

DOI: 10.1021/acs.jcim.9b00250
J. Chem. Inf. Model. 2019, 59, 2560−2571

Table 7. List of All False Subset Predictions during the Validation

pattern	supposed subset pattern
[+]	[S;D3](−N)(−[c,C])(−[c,C])
[+,++,+++]	[S;D3](−N)(−[c,C])(−[c,C])
[O+,o+,S+,s+]	[S;D3](−N)(−[c,C])(−[c,C])
[C+,Cl+,I+,P+,S+]	[S;D3](−N)(−[c,C])(−[c,C])
[SX3](*)	[S;D3](−N)(−[c,C])(−[c,C])
[#16v3,#16v5]	[S;D3](−N)(−[c,C])(−[c,C])
a1aaa2a(a1)aaa(a2):a	[n+]12nc3c4c5c(c3cc1c(nc(c2C)C)C)cccc5ccc4

N)(-[c,C])(-[c,C]) as a subpattern, and four of them contain positive charges. The disagreement is caused by the underlying NAOMI chemistry model. Although the model knows sulfur valence states with four explicit bonds, all of them are annotated with no neighboring hydrogen atoms. The compound ZINC000100315425 is out of the description domain of the chemistry model. In the pattern [S;D3](-N)(-[c,C])(-[c,C]), the sulfur node is expected to match sulfur with three single bonds and additional hydrogens. Thus, only positively charged sulfur atoms with three single bonds are assumed to match the pattern. Figure 11 shows all of the mentioned patterns and the ZINC compound. The relevant sulfur is encircled.

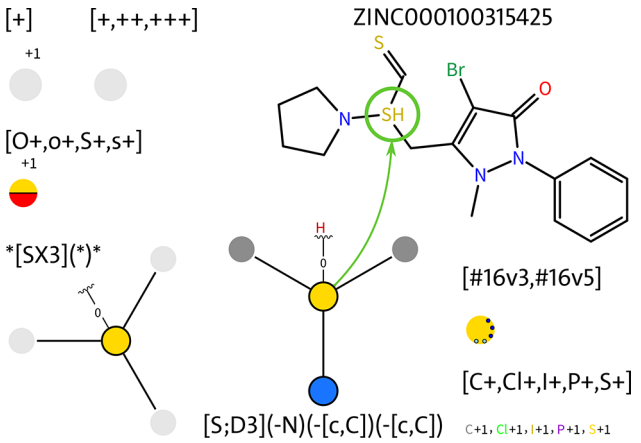


Figure 11. Six patterns that found [S;D3](−N)(−[c,C])(−[c,C]) as a subpattern that could not be experimentally validated, as ZINC structure ZINC000100315425 shows. The encircled sulfur has an explicit adjacent hydrogen, which is outside the model domain of the NAOMI chemistry model. There is no extended atom type representing a sulfur with four bonds and adjacent hydrogen. There is, however, an extended atom type for a positively charged sulfur with three bonds. According to the model, the sulfur should be positively charged and have only three single bonds.

The only remaining case in the first run is about a condensed aromatic pattern. ZINC structure ZINC000000386967 revealed that there may be small rings whose atoms are all aromatic but not every bond of this ring is aromatic, as shown in Figure 12. In this structure, a cycle of size 5 connects two polycyclic aromatic ring systems. The two bonds highlighted in green are not aromatic, although all five ring atoms are aromatic. This shows that the optional feature to detect bonds in rings consisting exclusively of aromatic atoms automatically as aromatic is not always right for SMARTS patterns.

A look at the remaining experimental subset pairs reveals many cases for which the validation data set is insufficient to detect the subtle differences between patterns. Often, certain fragments imply a constant reoccurring environment in the molecules. As long as other environments are theoretically possible, the experimental subset relationship is wrong.

This can be demonstrated with the generic carbonyl oxygen. Besides the typical atomtype for carbonyl oxygen with exactly one double bond, named 'O(010)', the chemistry model also contains an atomtype with a positively charged oxygen with a single and a double bond, named 'O(110)+'. The pyrylium ion justifies its existence. In the case of the generic carbonyl C=O, the model has to assume that the 'O(110)+' atomtype is possible even if there is no molecule where it occurs. Thus, patterns like C=O and C=[OX1] are not considered identical by the model, although one might expect this behavior.

Another common example is the identification of acyclic parts of the patterns. If a node or an edge of a pattern is explicitly acyclic, there are often several subset pattern pairs. For them the experimental subset relation is valid because there is no molecule in the validation data set for which the structure is part of a ring. Figure 13 shows a pattern pair combining both examples. The upper pattern describes two carbons, one of which is acyclic. In the experiment the same holds true for the two chain carbons in the lower left pattern. For SMARTScompare it is possible that the whole substructure is still part of a ring. Therefore, the singly bonded oxygen would be positively charged and have three bonds, as in oxonium. The modification of the pattern shown at the lower right enables subset detection by SMARTScompare. In conclusion, there are several experimentally valid subset relations that could be denied if there were molecules showing the pattern difference. This emphasizes the need for a compound-independent pattern comparison method like SMARTScompare.

CONCLUSION AND OUTLOOK

We have presented a novel solution to the generic chemical pattern comparison problem. The algorithm employing a maximum common subgraph (MCS) calculation is applicable to any substructure-based language for molecular patterns. It is inspired by a theoretical comparison approach for which patterns are identical if and only if their representations in chemical space are identical. A chemistry model is the foundation on top of which we designed a fingerprint descriptor for the pattern's nodes and edges. It is important that the fingerprint includes all available information on the node or edge and its environment in the pattern. Furthermore, redundancy in the pattern hampers correct comparison and has to be removed beforehand.

We implemented our algorithm for the widely used SMARTS language. The fingerprints cover most SMARTS features, including ring properties up to a ring size of 25. As a side effect, it is possible to detect common pattern errors resulting from incorrect valences or contradictions as well as redundant elements in the pattern. SMARTS recursion turns out to be a major difficulty in comparison of SMARTS patterns. The recursively defined environments usually describe more than one node, such that a truly recursive schema is unavoidable for its handling. As a proof of concept, the algorithm is implemented into the software tool SMARTScompare.

With SMARTScompare, it is possible to analyze SMARTS pattern pairs independent of their string representation. The algorithm is capable of similarity assessments and can detect subset relations between patterns. We validated the subset predictions of SMARTScompare using a large pattern collection applied to more than 370 million compounds. In that

DOI: 10.1021/acs.jcim.9b00250
J. Chem. Inf. Model. 2019, 59, 2560−2571

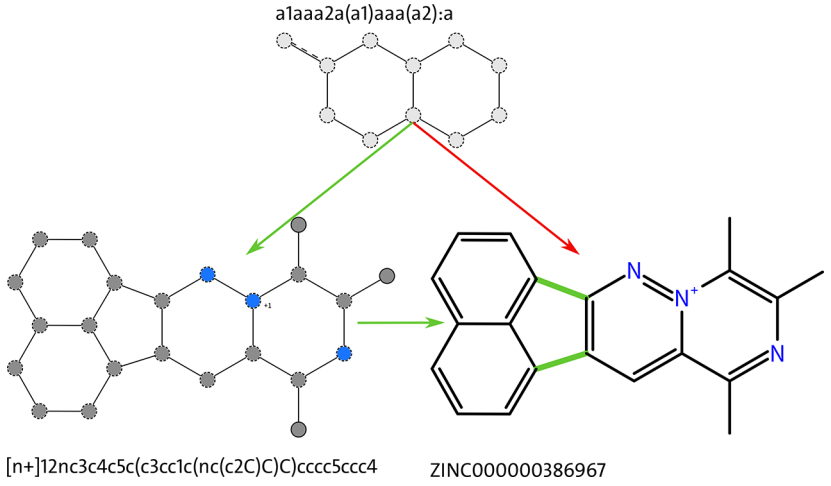


Figure 12. Two patterns in a subset relation in ZINC structure ZINC000000386967 that prohibits the subset detection in the validation. This structure shows that the detection of aromatic ring bonds can be erroneous.

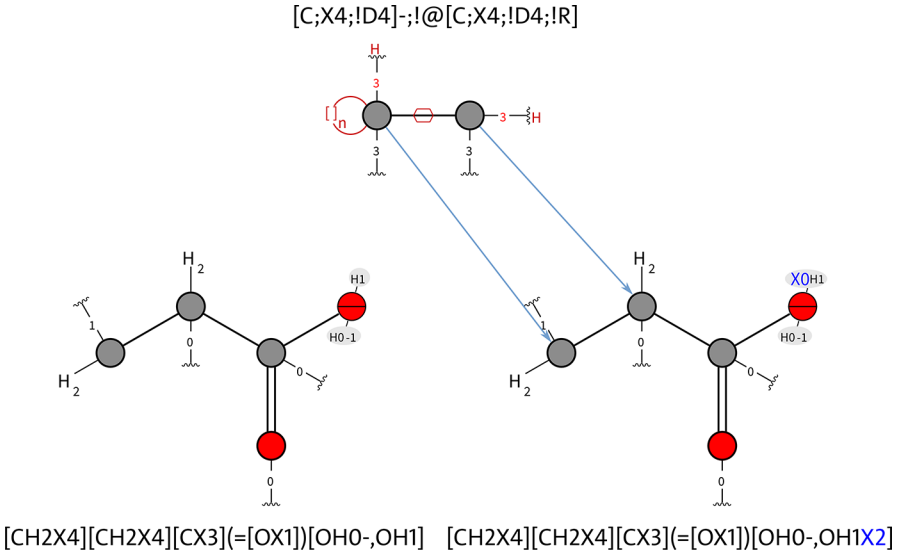


Figure 13. An experimentally valid pattern subset relation and the pattern modification such that SMARTScompare also detects it. Because of a seldom-used atomtype, it is still possible that the lower pattern is part of a ring. Upon application of a small modification, SMARTScompare also detects the subset relation.

experiment, SMARTScompare found thousands of experimentally valid pattern subset relations. The few observed issues merely affect the description domain of the underlying chemistry model and the handling of aromaticity. They have no impact on the correctness of the approach.

In a follow-up publication[27] we introduce a meaningful pattern similarity approach and an optional aromaticity detection method supporting the chemical intuition of aromaticity, which has to be explicitly modeled in SMARTS. We then demonstrate a showcase application of SMARTScompare by performing an in-depth similarity analysis of publicly available SMARTS pattern collections.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge on the ACS Publications website at DOI: 10.1021/acs.jcim.9b00250.

Additional chapters providing in detail descriptions of complex parts of the presented algorithm (AdditionalMethodChapters.pdf) and all of the SMARTS patterns that were used for the validation experiments (ValidationPatterns.smarts) (ZIP)

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: rarey@zbh.uni-hamburg.de. Phone: +49 (40) 428387351.

ORCID

Robert Schmidt: 0000-0002-7089-4842
Matthias Rarey: 0000-0002-9553-6531

Present Addresses

†F.O.: Evotec AG - Manfred Eigen Campus, Essener Bogen 7, 22419 Hamburg, Germany.
‡H.-C.E.: SAP SE - Innovation Center Potsdam, Konrad-Zuse-Ring 10, 14469 Potsdam, Germany.
§A.M.: Titel Media GmbH, Ritterstr. 9, 10969 Berlin, Germany.

DOI: 10.1021/acs.jcim.9b00250
J. Chem. Inf. Model. 2019, 59, 2560−2571

Author Contributions

H.-C.E., A.M., and M.R. developed the initial concept of atomtype-based SMARTS subset comparisons (first prototype, 2011). F.O. and M.R. extended the comparison concept toward similarity using a subgraph algorithm (second prototype, 2015). R.S. and M.R. developed the concepts for SMARTS recursion integration, fingerprint pruning, SMARTS node similarity, and MCS. R.S. implemented the SMARTScompare software based on the NAOMI cheminformatics library (current version). E.S.R.E. provided application scenarios for the SMARTScompare approach and analyzed and interpreted the results. R.S., E.S.R.E., and M.R. wrote the manuscript.

Notes

The authors declare the following competing financial interest(s): M.R. declares a potential financial interest in the event that the SMARTScompare software is licensed for a fee to non-academic institutions in the future.

SMARTScompare and SMARTScompareViewer, a tool for visualization of SMARTS relationships, are available for Linux, OS X, and Windows as part of the NAOMI ChemBio Suite (https://uhh.de/naomi) and are free for academic use and evaluation purposes. Furthermore, SMARTScompare and the SMARTScompareViewer are integrated in the SMARTSview server (https://smarts.plus). All feedback is greatly appreciated and supports the further development of SMARTScompare.

REFERENCES

(1) Daylight Chemical Information Systems, Inc. http://www.daylight.com/dayhtml/doc/theory/theory.smarts.html (accessed Aug 20, 2018).

(2) 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.

(3) 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.

(4) Gaulton, A.; et al. The ChEMBL database in 2017. *Nucleic Acids Res.* **2017**, *45*, D945−D954.

(5) Baell, J. B.; Holloway, G. A. 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*, 2719−2740.

(6) Bruns, R. F.; Watson, I. A. Rules for identifying potentially reactive or promiscuous compounds. *J. Med. Chem.* **2012**, *55*, 9763−9772.

(7) Rydberg, P.; Gloriam, D. E.; Zaretzki, J.; Breneman, C.; Olsen, L. SMARTCyp: A 2D method for prediction of cytochrome P450-mediated drug metabolism. *ACS Med. Chem. Lett.* **2010**, *1*, 96−100.

(8) Blake, J. F. Identification and evaluation of molecular properties related to preclinical optimization and clinical fate. *Med. Chem. (Sharjah, United Arab Emirates)* **2005**, *1*, 649−655.

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

(10) Pearce, B. C.; Sofia, M. J.; Good, A. C.; Drexler, D. M.; Stock, D. A. An empirical process for the design of high-throughput screening deck filters. *J. Chem. Inf. Model.* **2006**, *46*, 1060−1068.

(11) Brenk, R.; Schipani, A.; James, D.; Krasowski, A.; Gilbert, I. H.; Frearson, J.; Wyatt, P. G. Lessons learnt from assembling screening libraries for drug discovery for neglected diseases. *ChemMedChem* **2008**, *3*, 435−444.

(12) Sushko, I.; Salmina, E.; Potemkin, V. A.; Poda, G.; Tetko, I. V. ToxAlerts: A web server of structural alerts for toxic chemicals and compounds with potential adverse reactions. *J. Chem. Inf. Model.* **2012**, *52*, 2310−2316.

(13) Obrezanova, O.; Csányi, G.; Gola, J. M. R.; Segall, M. D. Gaussian processes: A method for automatic QSAR modeling of ADME properties. *J. Chem. Inf. Model.* **2007**, *47*, 1847−1857.

(14) Boström, J.; Falk, N.; Tyrchan, C. Exploiting personalized information for reagent selection in drug design. *Drug Discovery Today* **2011**, *16*, 181−187.

(15) Zhang, L.; Zhu, H.; Mathiowetz, A.; Gao, H. Deep understanding of structure-solubility relationship for a diverse set of organic compounds using matched molecular pairs. *Bioorg. Med. Chem.* **2011**, *19*, 5763−5770.

(16) Schärfer, C.; Schulz-Gasch, T.; Hert, J.; Heinzerling, L.; Schulz, B.; Inhester, T.; Stahl, M.; Rarey, M. CONFECT: Conformations from an expert collection of torsion patterns. *ChemMedChem* **2013**, *8*, 1690−1700.

(17) Guba, W.; Meyder, A.; Rarey, M.; Hert, J. Torsion Library Reloaded: A New Version of Expert-Derived SMARTS Rules for Assessing Conformations of Small Molecules. *J. Chem. Inf. Model.* **2016**, *56*, 1−5.

(18) Schomburg, K.; Ehrlich, H. C.; Stierand, K.; Rarey, M. From structure diagrams to visual chemical patterns. *J. Chem. Inf. Model.* **2010**, *50*, 1529−1535.

(19) Center for Bioinformatics Hamburg. SMARTSviewServer. https://smartsview.zbh.uni-hamburg.de (accessed Oct 11, 2018).

(20) Schomburg, K. T.; Wetzer, L.; Rarey, M. Interactive design of generic chemical patterns. *Drug Discovery Today* **2013**, *18*, 651−658.

(21) Borgelt, C.; Berthold, M. Mining molecular fragments: finding relevant substructures of molecules. *Proc. IEEE Int. Conf. Data Mining* **2002**, 51−58.

(22) Bietz, S.; Schomburg, K. T.; Hilbig, M.; Rarey, M. Discriminative Chemical Patterns: Automatic and Interactive Design. *J. Chem. Inf. Model.* **2015**, *55*, 1535−1546.

(23) Hopcroft, J. E.; Motwani, R.; Ullman, J. D. *Introduction to Automata Theory, Languages, and Computation*, 2nd ed.; Addison Wesley, 2001; Chapter 4, pp 125−168.

(24) Brüggemann-Klein, A.; Wood, D. One-Unambiguous Regular Languages. *Information and Computation* **1998**, *142*, 182−206.

(25) Hovland, D. The inclusion problem for regular expressions. *Journal of Computer and System Sciences* **2012**, *78*, 1795−1813.

(26) Landrum, G. RDKit: Open Source Cheminformatics. https://www.rdkit.org/ (accessed Sept 26, 2018).

(27) Ehmki, E. S. R.; Schmidt, R.; Ohm, F.; Rarey, M. Comparing Molecular Patterns using the Example of SMARTS: Applications and Filter Collection Analysis. *J. Chem. Inf. Model.* **2019**, DOI: 10.1021/acs.jcim.9b00249.

(28) 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.

(29) Zamora, A. An Algorithm for Finding the Smallest Set of Smallest Rings. *J. Chem. Inf. Model.* **1976**, *16*, 40−43.

(30) Kolodzik, A.; Urbaczek, S.; Rarey, M. Unique ring families: A chemically meaningful description of molecular ring topologies. *J. Chem. Inf. Model.* **2012**, *52*, 2013−2021.

(31) Bron, C.; Kerbosch, J. Algorithm 457: finding all cliques of an undirected graph. *Commun. ACM* **1973**, *16*, 575−577.

(32) Koch, I. Enumerating all connected maximal common subgraphs in two graphs. *Theor. Comput. Sci.* **2001**, *250*, 1−30.

(33) Ehrlich, H. C.; Rarey, M. Maximum common subgraph isomorphism algorithms and their applications in molecular science: A review. *Wiley Interdiscip. Rev.: Comput. Mol. Sci.* **2011**, *1*, 68−79.

(34) McCreesh, C. Parasols. https://github.com/ciaranm/parasols (accessed Sept 26, 2018).

(35) ZINC15. https://zinc15.docking.org (accessed Aug 20, 2018).

(36) Ehrlich, H. C.; Rarey, M. Systematic benchmark of substructure search in molecular graphs - From Ullmann to VF2. *J. Cheminf.* **2012**, *4*, 13.

DOI: 10.1021/acs.jcim.9b00250
J. Chem. Inf. Model. 2019, 59, 2560−2571