



The
University
Of
Sheffield.

Applications and Variations of the Maximum Common Subgraph for the Determination of Chemical Similarity

by **Edmund Duesbury**

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of
Philosophy

The University of Sheffield
Faculty of Social Sciences
Information School

October 2015

Acknowledgements

I first wish to give thanks to my supervisors, Professor Peter Willett and Doctor John Holliday, for their valuable input and guidance. This thesis would not have been possible without them. I extend my thanks to several members of the chemoinformatics research group. My colleague James Wallace, whose valuable information on KNIME and Java helped me with my work, particularly when getting started. Matthew Seddon also, for his knowledge on mathematics. Jorge Valencia, for a number of interesting nights and discussions. Nor Sani, Lucyantie Mazalan and Hasmila Omar I thank, for their valuable contributions of biscuits and chocolates to aid my cognitive functioning. Other members of the chemoinformatics research group I give my thanks to, include Christos Kannas, Val Gillet, Hua Xiang, Sonny Gan, Richard Sherhod and Eleanor Gardiner.

Several others gave support which aided the work of this project, including

- George Papadatos and Stephan Beisken, for introducing me to KNIME and KNIME development
- Basil Hartzoulakis, for inspiring me to pursue a path in chemoinformatics
- Eleanor Gardiner and Dave Cosgrove, for their valuable input with an implementation of RASCAL
- John May, for much advice on usage and support with the Chemistry Development Kit
- Yuanyuan Zhu and Professor Jeffrey Yu, for additional information on their graph matching algorithm consR
- Andrew Dalke, for implementation details and explanations on his algorithm fMCS
- Peter Englert, for invaluable information in the usage of ChemAxon's maximum common substructure algorithm
- Hua Xiang, for information on chemoinformatics datasets as well as weighting schemes
- Andreas Bender, for some insightful discussions within the discipline of chemoinformatics

I would also like to give thanks to the Shodokan Aikido dojo in Sheffield, including (but not limited to) instructors Scott Allbright and Celine Pagnier, particularly for their help in my progression to shodan in Aikido, as well as teaching experience in non-academic matters.

Finally I am grateful for the support of my family, including my parents, my sister and my grandmother Zona, which kept me going throughout my time at Sheffield. I look forward to working with Will Pitt, along with my future colleagues, at the chemoinformatics division at UCB Celltech, the position of which I could not have obtained without the help of my doctoral supervisors.

Abstract

The Maximum Common Substructure (MCS), along with numerous graph theory techniques, has been used widely in chemoinformatics. A topic which has been studied at Sheffield is the hyperstructure concept - a chemical definition of a superstructure, which represents the graph theoretic union of several molecules. This technique however, has been poorly studied in the context of similarity-based virtual screening. Most hyperstructure literature to date has focused on either construction methodology, or property prediction on small datasets of compounds.

The work in this thesis is divided into two parts. The first part describes a method for constructing hyperstructures, and then describes the application of a hyperstructure in similarity searching in large compound datasets, comparing it with extended connectivity fingerprint and MCS similarity. Since hyperstructures performed significantly worse than fingerprints, additional work is described concerning various weighting schemes of hyperstructures.

Due to the poor performance of hyperstructure and MCS screening compared to fingerprints, it was questioned whether the type of maximum common substructure algorithm and type had an influence. A series of MCS algorithms and types were compared for both speed, MCS size, and virtual screening ability. A topologically-constrained variant of the MCS was found to be competitive with fingerprints, and fusion of the two techniques overall improved active compound recall.

Contents

Contents	5
Abbreviations	9
1 Introduction	15
2 Hyperstructures and Chemical Graphs	19
2.1 Overview	19
2.2 Graph Theory and Chemical Structures	19
2.2.1 Introduction	19
2.2.2 Graph Theory Terminology	21
2.2.3 Representation of Chemical Structures	24
2.3 Substructure Searching	34
2.3.1 Subgraph Isomorphism Algorithms	36
2.4 Finding the Maximum Common Subgraph	37
2.4.1 Application of Cliques to Find the MCS	39
2.4.2 MCS Categories - Modifying the Modular Product	41
2.4.3 Clique Detection with Graph Colouring	44
2.4.4 Clique Detection Algorithms and the MCS	45
2.4.5 cMCS via Subgraph Enumeration	56
2.4.6 Other MCS Algorithms	60

2.5	Chemical Hyperstructures	64
2.5.1	Development of Definitions and Properties of Hyperstructures	64
2.5.2	Construction of Hyperstructures	67
2.5.3	Predictive Applications of Hyperstructures	70
2.5.4	Caveats with 3D Methods	74
2.6	Conclusions	75
3	Similarity Measures in Chemoinformatics	77
3.1	Introduction	77
3.2	What is Similarity?	77
3.3	Weighting Schemes	80
3.4	Similarity Coefficients	81
3.4.1	Symmetric coefficients	82
3.4.2	What is the Best Symmetric Coefficient?	84
3.4.3	Asymmetric Coefficients	85
3.5	Graph Theory Similarity	87
3.6	The Activity Landscape Paradigm	91
3.7	Data Fusion	94
3.7.1	Why Data Fusion is Believed to Work	97
3.8	Conclusion	99
4	Construction and Virtual Screening of Hyperstructures	101
4.1	Introduction	101
4.2	Materials and Methods	101
4.2.1	Hardware and Software	101
4.2.2	Compound Data Sets	102
4.2.3	Maximum Common Substructure	104

4.2.4	Hyperstructure Construction and Application	105
4.2.5	Similarity Searching	106
4.2.6	Evaluation Metrics	107
4.3	Results and Discussion	112
4.3.1	Virtual Screening Recall Performance	112
4.3.2	Diversity Performance	117
4.3.3	Relative Performances	120
4.3.4	Method Complementarity	122
4.3.5	Time and Compression Results	123
4.3.6	Similarity Fusion	126
4.3.7	Variability	128
4.3.8	Physicochemical Properties of Retrieved Compounds	128
4.4	Conclusions	129
5	Similarity Searching with Modified Hyperstructures	131
5.1	Introduction	131
5.2	Materials and Methods	131
5.2.1	Monitoring Ghost Substructures	132
5.2.2	Bond Weighting	133
5.2.3	Topology Type Matching	137
5.3	Results and Discussion	138
5.3.1	Virtual Screening Recall Performance	138
5.3.2	Ghost Substructure Information	142
5.4	Conclusions	143

6	Evaluation of MCS algorithms for Alignment and Virtual Screening	145
6.1	Introduction	145
6.2	Materials and Methods	146
6.2.1	A Note on Implementation	146
6.2.2	Modular Product of Two Graphs	147
6.2.3	Modular Product Simplification Heuristics	148
6.2.4	Compound Datasets	149
6.2.5	Benchmarks	162
6.3	Results and Discussion	164
6.3.1	S, N and M Compound Pairs	164
6.3.2	Franco Compounds	169
6.3.3	Virtual Screening	178
6.4	Conclusions	182
7	Conclusions	183
7.1	Summary	183
7.1.1	MCS-based Hyperstructure Construction	183
7.1.2	Evaluation of MCS Algorithms and Hyperstructures in Similarity Searching	183
7.1.3	Comparing Virtual Screening Ability of MCS techniques with Fingerprints	184
7.2	Limitations	185
7.3	Recommendations for Future Investigations	186
7.3.1	Weighted MCS	187
7.3.2	Improvement of Clique Detection Algorithm	187
7.3.3	Hyperstructure Improvements	188
7.4	Final Remarks	192
	References	193

Edmund Duesbury	Contents
Appendix A Hyperstructure Construction Algorithm	219
Appendix B Diversity Statistics Bar Charts	223
Appendix C Similarity Search Method Complementarity	231
Appendix D Tabulated Results of Search Methods	239
Appendix E Summary performances of Search Methods	245
Appendix F Mean Ranks of Methods	247
Appendix G Activity Class performance on Weighting Schemes	251
Appendix H Modular Product Heuristics	255
Appendix I MCS algorithm implementation information	263
Appendix J Modular Product Information	269
Appendix K S, N and M Compound Pair Performances	273
Appendix L Franco Compound Pair Performances	277

Abbreviations

AUROC Area Under the Receiver Operating Characteristic

AWCH Activity-Weighted Chemical Hyperstructure

BEDROC Boltzmann-Enhanced Receiver Operating Characteristic

CAS Chemical Abstracts Service

CDK Chemistry Development Kit

cMCES connected Maximum Edge Induced Subgraph

CoMFA Comparative Molecular Field Analysis

dMCES disconnected Maximum Edge Induced Subgraph

EF Enrichment Factor

FRELS Fragments Reduced to an Environment that is Limited

GA Genetic Algorithm

GALAHAD Genetic Algorithm with Linear Assignment of Hypermolecular Alignment of Datasets

GMM Gaussian Mixture Model

InChI International Chemical Identifier

IR Information Retrieval

KNIME Konstanz Information Miner

MCES Maximum Common Edge Subgraph

MCIS Maximum Common Induced Subgraph

MCS Maximum Common Subgraph

MDDR MDL Drug Data Report

MFTA Molecular Field Topology Analysis

MiCS Minimum Common Supergraph

MIF Molecular Interaction Fields

MMP Matched Molecular Pair

MOS maximum overlap set

MPS Mean Pairwise Similarity

MTD Minimal Topological Difference

MUV Maximum Unbiased Validation

NP-complete Non-Deterministic polynomial time Complete

PCA Principal Component Analysis

PDB Protein DataBank

PLS Partial Least Squares

RASCAL Rapid Similarity Calculation (algorithm)

ROC Receiver Operating Characteristic

SALI Structure-Activity Landscape Index

SAR Structure-Activity Relationship

SAS Structure-Activity Similarity (plot)

SDF Structure-Data File

SEA Similarity Ensemble Approach

SLN Sybyl Line Notation

SMILES Simplified Molecular Input Line Entry Specification

tdMCS topologically-constrained dMCS

TSS Turbo similarity searching

WLN Wiswesser Line Notation

WOMBAT World of Molecular Bioactivity

Chapter 1

Introduction

Similarity in general is a highly subjective topic, which manifests differently depending on the task at hand. In the discipline of chemoinformatics, molecules are often represented as graphs, and graph theory has become a popular method for deriving hypotheses for chemical phenomena. An intuitive method for calculating similarity involves the Maximum Common Subgraph (MCS), often referred to in chemoinformatics as the Maximum Common Substructure. Briefly, the MCS can be thought of as the intersection or overlap of two graphs, otherwise showing features shared between each graph. Due to the inherently NP-complete nature of the MCS detection problem, extensive research has been performed to yield accurate solutions in feasible time frames. Recent research has presented methods that make it possible to perform similarity searching of large compound datasets. This thesis focuses on using the MCS to search compound datasets, as well as to calculate the graphical unions of molecules, referred to as hyperstructures, which to date have not been examined for searching large datasets of compounds. Importantly, this thesis has three underlying objectives:

- To create MCS-based method(s) for the construction of hyperstructures
- To evaluate the computational feasibility of various MCS types, algorithms, and the hyperstructure concept in similarity searching of large datasets, typically in excess of 100 000 compounds.
- To compare the MCS-based methods with molecular fingerprints - an industry standard for similarity-based virtual screening

Chapter 2 introduces the reader to the terms of graph theory relevant to the work being carried out in this thesis. This is extended to chemical graph theory, describing what a chemical graph is, and

outlining common methods of representing molecules, including the widely used types of molecular fingerprint that are used for fast similarity searching. Literature on the MCS and hyperstructures is then introduced, focusing on MCS computation and hyperstructure calculation. Some applications of hyperstructures will be discussed, most of which are outside the domain of similarity searching.

Chapter 3 defines similarity searching in both graph theoretical and non-graph theoretical terms. A diverse set of literature is discussed here, concentrating on similarity equations used in conventional database searching. The activity cliff concept is introduced in this chapter, highlighting differences between chemical similarity types. Some similarity visualisation methods (such as activity landscape visualisation) are briefly covered. Data fusion is also overviewed here, detailing the types of data fusion, and their uses in virtual screening.

Chapter 4 introduces the main body of experimental work concerning hyperstructures. This chapter aims to address aspects of all three objectives, accomplishing the first objective of an MCS-based hyperstructure construction method, in addition to examining and comparing its speed and recall in similarity-based virtual screening of three large chemical datasets. Here, the hyperstructure concept, an inexact MCS method (via data fusion), and a high performance molecular fingerprint data fusion method are evaluated for virtual screening. Details of the similarity searching, including a substructure similarity-based method for hyperstructures, are detailed here. The main conclusion from this chapter is that the fingerprint method is the fastest and most powerful virtual screening method, though fusion of the three techniques yields interesting results, and all three techniques complement each other in the types of compounds retrieved. We also show the benefits of using an asymmetric similarity coefficient when concerning hyperstructures, compared to traditional symmetric similarity coefficients. Much of the work in this chapter was recently published as an article (Duesbury, Holliday, & Willett, 2015).

Chapter 5 explores the second objective further, where recent work on weighting schemes at Sheffield is applied to hyperstructure similarity searching. Objective three is also re-explored, as fingerprints are compared with the weighting methods. Five different weighting schemes are used on hyperstructures whose edges are weighted, based on the frequency of overlap with the molecules that the hyperstructure was trained on. Additional influences on the topology types found in molecules (i.e. rings and chains) are assessed via the weighting schemes. The presence of artefacts referred to as “ghost substructures” - features present in the hyperstructure that are absent from its training molecules - are monitored throughout the searches.

Chapter 6 focuses on objective two, although purely on MCS algorithms as opposed to hyperstructures. Objective three is also re-explored, as fingerprints are compared with the MCS methods. This chapter evaluates various MCS algorithms and MCS types in similarity searching, in contrast to Chapter 4 which uses one MCS algorithm for two graph-based techniques. To date, no extensive comparisons of MCS algorithms have been performed in chemoinformatics since the turn of the millennium. This work continues a comparison of clique-based MCS algorithms carried out at Sheffield in 1997, using both recent algorithms, as well as older algorithms which have been proven to have high performances. This chapter presents three benchmarks, notably a set of compounds which are challenging for exact MCS algorithms to compute in a short space of time. The virtual screening performance of a subset of the algorithms and MCS types is also assessed, with particular attention paid to whether an exact MCS is actually needed for satisfactory virtual screening, as opposed to a viable and significantly faster inexact solution. This chapter also gives some insight into the relevance of the tested MCS types to biological activity.

Chapter 7 concludes the thesis, offering some extensions to the work carried out on hyperstructure and MCS similarity searching, as well as highlighting the main conclusions and worth of the research carried out.

Chapter 2

Hyperstructures and Chemical Graphs

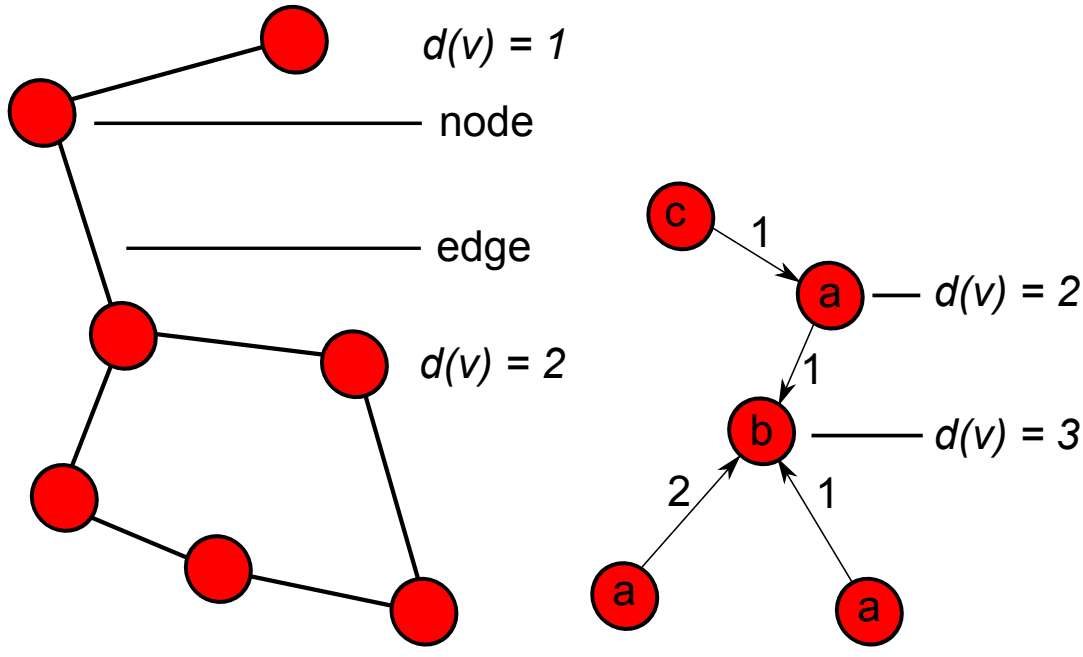
2.1 Overview

The molecular hyperstructure is a somewhat neglected concept which originated not from graph theory, but from chemistry itself. The hyperstructure has a distinct set of properties that makes it potentially attractive for a number of applications in the discipline of chemoinformatics. This chapter overviews the concept of graph theory, in both chemical and mathematical terms. A selection of methods for analysing graphs is introduced, mostly with a focus on chemical graphs. The subgraph problem is then introduced, along with methods for analysing and finding subgraphs. This then leads onto the hyperstructure concept, which is introduced first with a description of its history, along with details on methods for constructing, searching with, and applications of hyperstructures.

2.2 Graph Theory and Chemical Structures

2.2.1 Introduction

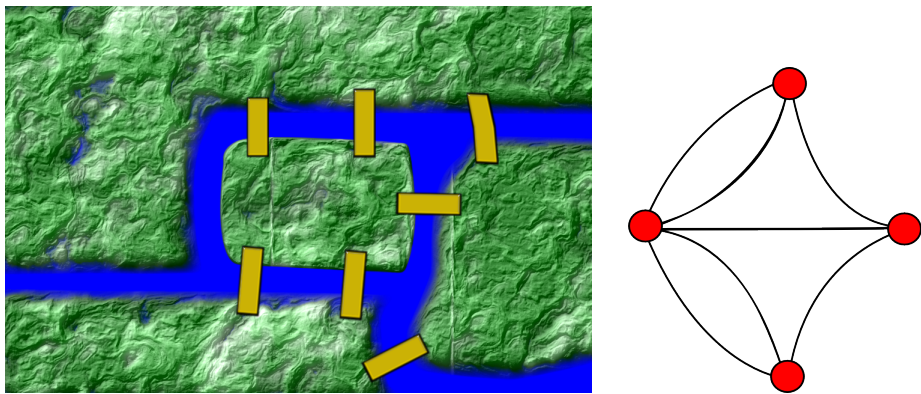
Ever since the first widely accepted publication was presented by Leonhard Euler in 1736 on the *Seven Bridges of Königsberg*, graph theory has developed into an independent subject. Euler's paper presented the idea of attempting to visit the four land masses of the city by crossing each of the seven bridges exactly once. In abstracting this problem as a graph, the problem is more easily visualised. Simply put, a graph is a set of *vertices* or nodes, that may or may not be connected to other vertices by *edges* or arcs (Figure 2.1). In Euler's example, the vertices represent the four land masses, whereas the edges represent the bridges (Figure 2.2) (Diestel, 2006; Euler, 1741).



(a)

(b)

Figure 2.1. a) is a graph where the nodes, edges and degree are indicated (for instance, the node with a degree of 1 has its degree as it has only one edge incident to it). b) is a directed graph with arbitrarily assigned node labels, and edge weights.



(a)

(b)

Figure 2.2. A representation of the seven bridges of Königsberg (a), and in graph form (b).

Graph theory has been used to abstract many problems in the real world, both within and without chemistry. This not only simplifies the problem domain, but also enables the application of graph matching algorithms to more feasibly solve a problem (some of which will be explained later in this chapter). An old example is the travelling salesman problem, whereby one wishes to find the shortest distance to travel between a set of points. Graph theory has been applied to solve this problem (Kruskal, 1956), and has also been applied to several others, including the analysis of the World Wide Web, in which nodes represent web pages and edges represent hyperlinks (Broder et al., 2000), and

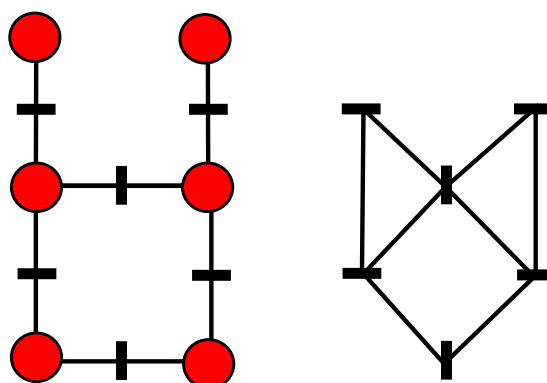


Figure 2.3. A graph with its edges marked (left), and its line graph (right)

to address problems in natural language processing and artificial intelligence (Navigli & Ponzetto, 2012).

2.2.2 Graph Theory Terminology

Mathematically a graph, G , can be written as $G = (V, E)$, where V and E represent the vertices and edges respectively. An edge in graph theory is where two vertices are connected, or *adjacent* to each other, thus the connected vertices v_1 and v_2 form an edge so that $(v_1, v_2) \in E(G)$. $E(G)$ and $V(G)$ represent the edge and vertex sets in a graph, respectively. The *degree* of a node is the number of edges it possesses (Figure 2.1b). Note that a node may be connected to itself, which also counts towards the degree of the node (though this phenomenon does not occur in chemical graphs). For an undirected graph with a vertex count of $|V|$ and an edge count of $|E|$, the *graph density* is defined as

$$D = \frac{2|E|}{|V|(|V| - 1)}$$

thus, a dense graph would be one with a high level of connectivity - more nodes would be connected between each other, than in a sparse graph. A graph which possesses nodes with descriptors is classed as a *labelled* graph. Likewise, when edges have descriptors, a graph is considered *weighted* (Figure 2.1b). An example of a labelled weighted graph is the map of the London underground, where the nodes represent each tube stop, and the edges are described by colour, each colour signifying a different route. A *directed graph* has edges which point from one node to another. A flow diagram could be thought of as a directed graph, for each edge directs a node to the next node in sequence.

A *line graph* is a graph which is constructed from the edges of an existing graph. Thus, an edge in

a graph G is converted to a vertex in $L(G)$. Two vertices are connected in $L(G)$ if the corresponding edges in G are connected to a common vertex (Whitney, 1932). An example of line graph derivation is shown in Figure 2.3.

A *complement graph* \bar{G} is where given a graph $G = (V, E)$, the complement graph contains the inverse of the edges, thus $\bar{G} = (V, \bar{E})$. In short, if one views a graph's edge set as an adjacency matrix (explained later), the complement set of edges \bar{E} is where each element of the adjacency matrix is inverted (0 is set to 1, and *vice versa*) (Bomze, Budinich, Pardalos, & Pelillo, 1999).

2.2.2.1 Graph Isomorphism. Two graphs G_1 and G_2 are *isomorphic* if there is a bijective, or one-to-one mapping of vertex sets $V_1 \rightarrow V_2$, and a likewise bijective mapping of edges $E_1 \rightarrow E_2$. Simply put, a graph is isomorphic to another graph if their topologies are identical. Figure 2.4 shows an example of graph isomorphism (Diestel, 2006). Isomorphism between two graphs can be determined via the use of matrix algebra. This however relies on eigendecomposition, the basic mathematics of which can be read in (A. R. Leach & Gillet, 2007a).

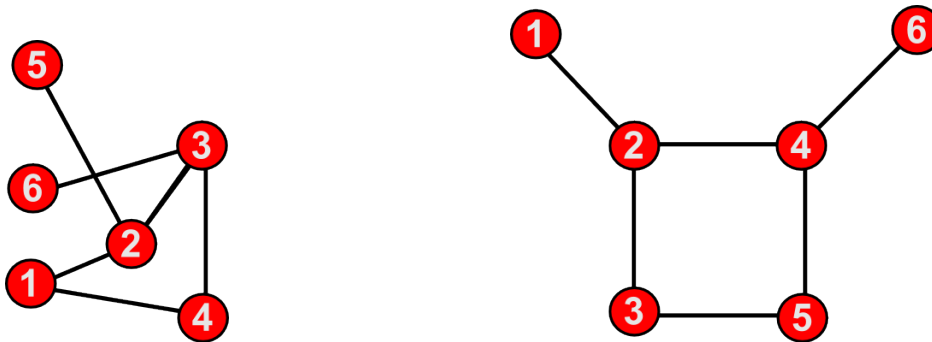


Figure 2.4. An example of graph isomorphism. Identically numbered nodes in the two graphs correspond to each other.

2.2.2.2 Subgraphs and Supergraphs. A *subgraph* of graph G is a graph G' such that $G' \subseteq G$, thus possessing a smaller set of the vertices and edges of the parent graph. An *induced subgraph* is a subgraph G' of a graph G where all edges connecting the used vertices V' in G' are also present in G . An *edge-induced subgraph* by contrast is a set of edges taken from the parent graph, in which vertices incident (connected to) to the edges are included. A subgraph is a *common subgraph* of graphs G_1 and G_2 if it is isomorphic to the subgraphs G'_1 and G'_2 of G_1 and G_2 respectively. The *Maximum Common Induced Subgraph* (MCIS, Figure 2.5c) is the largest (in terms of vertices) induced subgraph common to G_1 and G_2 , whereas the *Maximum Common Edge Subgraph* (MCES), is the largest number of edges isomorphic to G_1 and G_2 (Diestel, 2006; Raymond & Willett, 2002b). The MCES can be further abstracted into the *connected* (cMCES) and *disconnected* MCES (dMCES). The cMCES

(Figure 2.5d) is the single MCES graph, where all the nodes in the subgraph are connected to at least one other node in the subgraph. The dMCES (Figure 2.5e) by contrast, sometimes referred to the maximum overlap set (MOS), can contain multiple (separated) subgraphs, representing all the edges in common between the graphs being matched. It is important to distinguish between the MCIS and the MCES. From this point and onwards in this thesis, the term *Maximum Common Subgraph* (MCS) will be used if the distinction between MCIS and MCES is not important. As will be shown later in this chapter, the MCS is an important concept in chemistry as it forms the basis for graph-based similarity searching, as well as a method for constructing hyperstructures.

In order to understand the basis of what hyperstructures are, the concept of *supergraphs* must be introduced. A supergraph quite simply is the converse of a subgraph, being that the parent graph of a subgraph is its supergraph. A *common supergraph* G_S of two graphs G_1 and G_2 is where there exist subgraph isomorphisms from G_1 to G_S , and G_2 to G_S . The *Minimum Common Supergraph* (MiCS) likewise is the common supergraph between the two graphs with the fewest nodes and edges, of all possible common supergraphs (Figure 2.5f) (Bunke, Jiang, & Kandel, 2000). A simple contrast between the MCS and the MiCS, is to think of the MCS as $G_1 \cap G_2$, being the boolean AND, or intersection of vertices and edges between two graphs. The MiCS in contrast could be thought of as $G_1 \cup G_2$, being the boolean OR, or the union of the vertices and edges of the two graphs. This is not strictly true, as some kind of mapping would be required to assign vertices from G_1 to G_2 , otherwise the union would simply be $V_1 + V_2$. More correctly, the MiCS would be the union of the MCS and the *difference* graph (i.e. $G_2 - G_1$), where the difference and MCS are connected via a set of edges formally referred to as the *embedding* (Bunke et al., 2000).

In this thesis, the term *compression* will be used to refer to the relationship of superstructure (and thus hyperstructure) size, to the cumulative size of the graphs used to build the superstructure. A more stringent, or “better” compression, refers to a relatively small hyperstructure given a set of unique input graphs. This concept will be discussed in more detail later in this chapter.

2.2.2.3 Cliques. A *clique* in graph theory is a series of vertices such that each vertex is connected to each other vertex. A *maximal clique* in a graph is a clique that cannot be extended - no nodes can be added to it which can enlarge the clique (whilst keeping it a clique). A *maximum clique* of a graph is a maximal clique of the largest possible size (in terms of nodes), bringing the possibility that a graph can in fact possess multiple maximum cliques. Clique detection algorithms make up a group of methods which are important for finding the MCS of graphs, some of which will be explained later in the chapter. A triangle is an example of a clique, for all the nodes are connected to each other

(Raymond & Willett, 2002b).

2.2.2.4 Vertex Covers. Given a graph G with vertices V and edges E , A vertex cover C is a subset of vertices such that for all edges $(u, v) \in E$, $u \in C$ or $v \in C$. Thus, it is a set of vertices which “includes” all the edges in the graph, being that for each edge in the graph G , there is at least one vertex in the cover which is adjacent to said edge. In analogy to clique terminology, a *minimal vertex cover* is one where no vertices can be removed from a cover without losing edges (Figure 2.5g), and a *minimum vertex cover* is a vertex cover of the smallest size in the graph. A related concept is that of an *independent set*, which is a set of vertices in which no vertex is adjacent to another in the set. A maximum independent set is actually the opposite set of vertices to its corresponding minimum vertex cover (that is, subtracting a minimum vertex cover from a graph will leave a maximum independent set). Of greater relevance to the work of this thesis, is that the maximum independent set of a complement graph \bar{G} corresponds to the maximum clique in the graph G . Likewise, given a minimum vertex cover C in \bar{G} , the corresponding maximum clique in G is $V - C$ (Bomze et al., 1999).

2.2.2.5 Median Graph. The *median graph* concept represents the centroid of a set of graphs. Nebeský (1971) was the first to mention the term “median graph” in the literature, defining first a median vertex as the centre between three vertices, then the median graph itself as a graph where all triplets of vertices in the graph have just one median. By contrast, Jiang, Müunger, and Bunke (2001) defined a median graph g as the most representative graph of a graph set S with N graphs, in this case by having the smallest sum of distances of its nodes to other nodes of other graphs. This can be shown as $g \in S$ given that the distance m of g to the members of set S follows the condition

$$m = \min\left(\sum_{i=1}^N d(g, g_i)\right)$$

where $d(g, g_i)$ is the distance from the median graph to graph i from set S . Previous applications of the median graph include the classification of graphs (Ferrer, Valveny, Serratos, Riesen, & Bunke, 2010), optical character recognition (Jiang et al., 2001) and image clustering (Hlaoui & Wang, 2006).

2.2.3 Representation of Chemical Structures

Chemical compounds were represented by graphs for a long time before the discipline of chemoinformatics existed. The first recorded application of graph theory in chemistry was by Cayley (1857), who used graph theory in an attempt to represent all the forms of paraffin with five atoms, the carbon atoms being nodes with the chemical bonds as the edges. However, it was Sylvester (1878) who

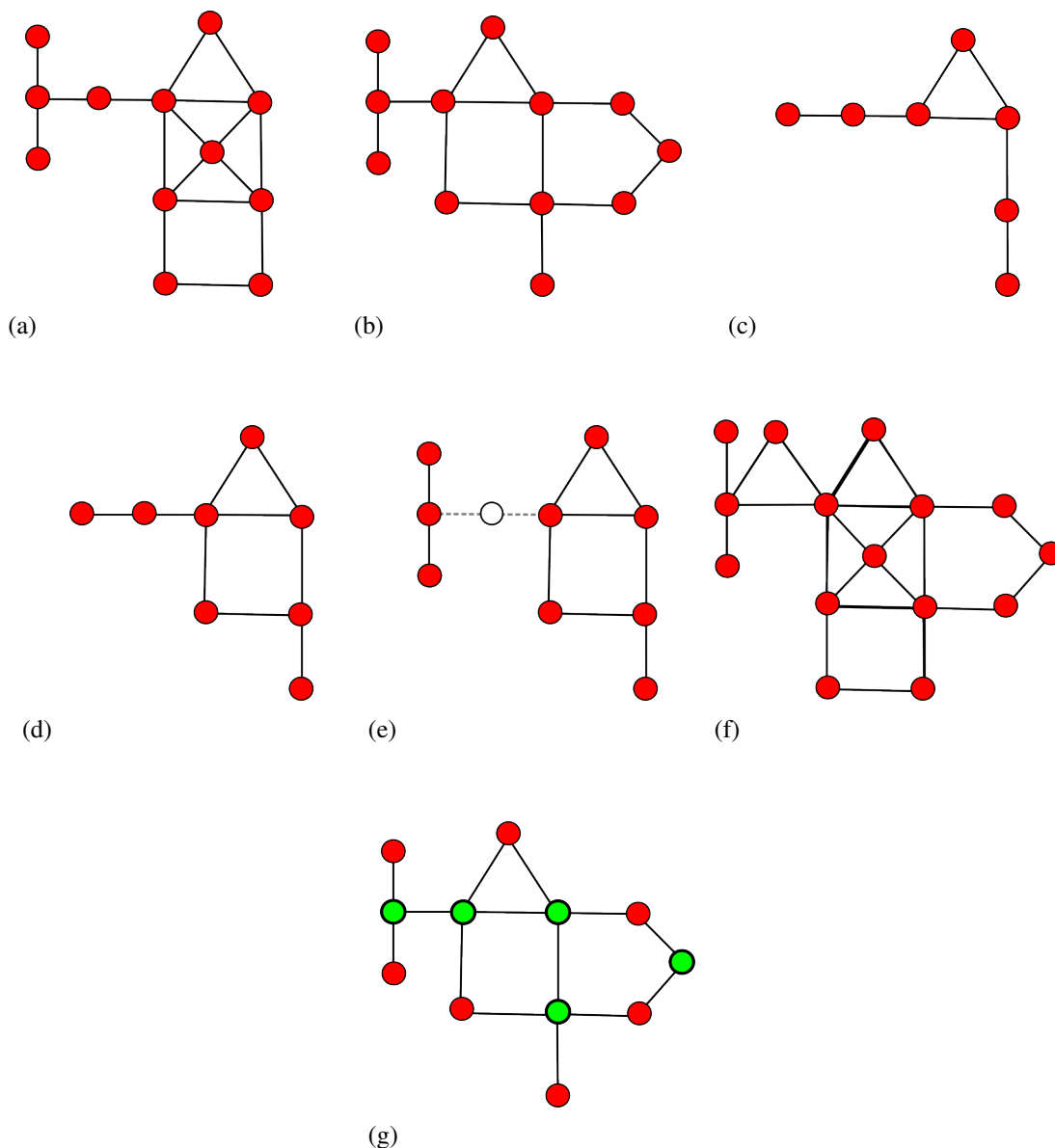


Figure 2.5. 2.5a is G_1 whereas 2.5b is G_2 . 2.5c is the MCIS, 2.5d is the cMCES of G_1 and G_2 , and 2.5e is the dMCES. 2.5f is the MiCS. Note how the MCES has a greater coverage of G_1 and G_2 compared to the MCIS. 2.5g is a minimal vertex cover of G_2 (The cover vertices being highlighted in green).

generalised this in mathematical terms of graph theory, referring at one point directly to Cayley's work. Since these publications there are now three main methods of representing chemical graphs computationally to date, these being linear notations, connection tables, and molecular descriptors.

2.2.3.1 Connection Tables. Also known as a coordinate list (May & Steinbeck, 2014), an early method of rigorously defining compounds was the *connection table* (also known as chemical table file). One of the first publications showing a format reminiscent of modern connection tables is by

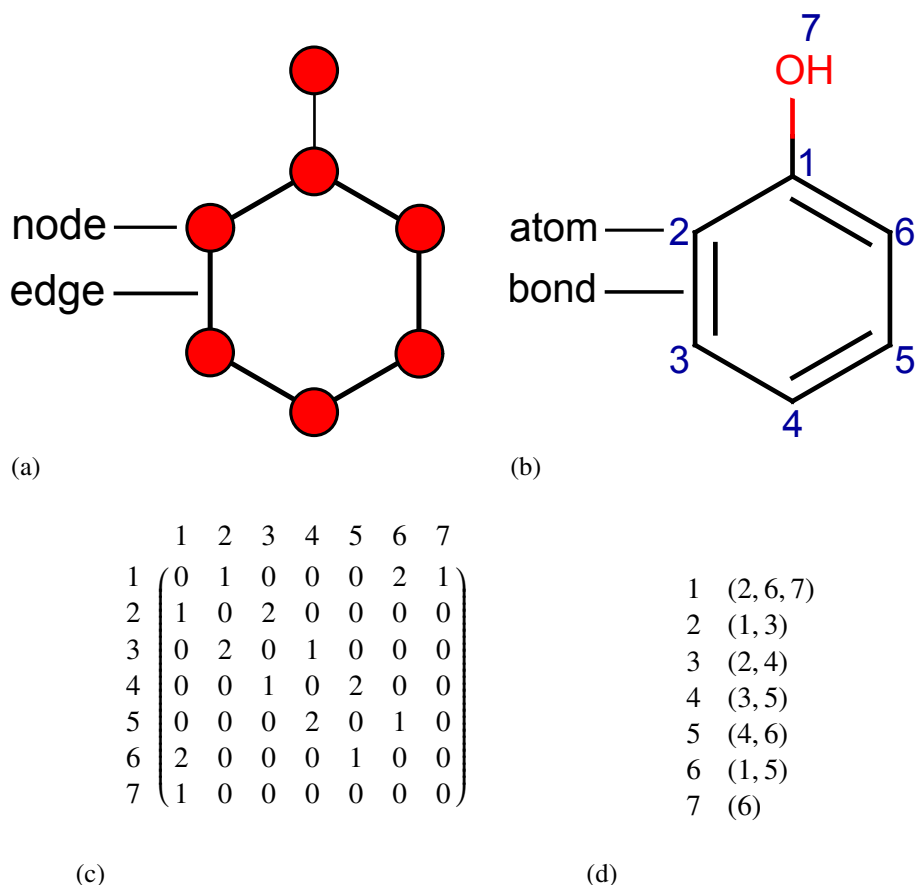


Figure 2.6. Graph representations of the compound phenol. a) is an example of a graph, and b) is the weighted (bond type) and labelled (atom type) version of that graph to represent phenol. c) is the adjacency matrix of phenol, which depicts which nodes (row-wise) are bound to which other nodes (column-wise). In this example, an edge's weight is displayed (1 or 2 representing a single or double bond respectively), and no edge is identified as a 0, rather than the binary statement of being adjacent or not. d) is the adjacency list of phenol, where for a given row (or node identifier), the adjacent node identifiers are given in brackets.

Dyson, Cossum, Lynch, and Morgan (1963). Initially, they converted a molecular graph to a "cipher" depicting a linear arrangement of the atoms with relevant bond types. The "cipher" was then assembled into what was referred to as an "extended matrix," a simple form of connection table which described which atoms were connected to which other atoms in the molecule, and the respective bond types. Perhaps surprisingly, Hiz (1964) also presented an implementation involving cipher matrices (apparently being unaware of the work done at CAS), though the cipher was assembled into an adjacency matrix (Figure 2.6c) to describe which atoms (including hydrogens) were connected to each other (and by what nature of bond). One of the most widely used connection table formats now is the MDL MOLFile format, an example of which is shown in Figure 2.7b. The first three lines of the format represent the header, which often stores the molecule's name. The next line represents the

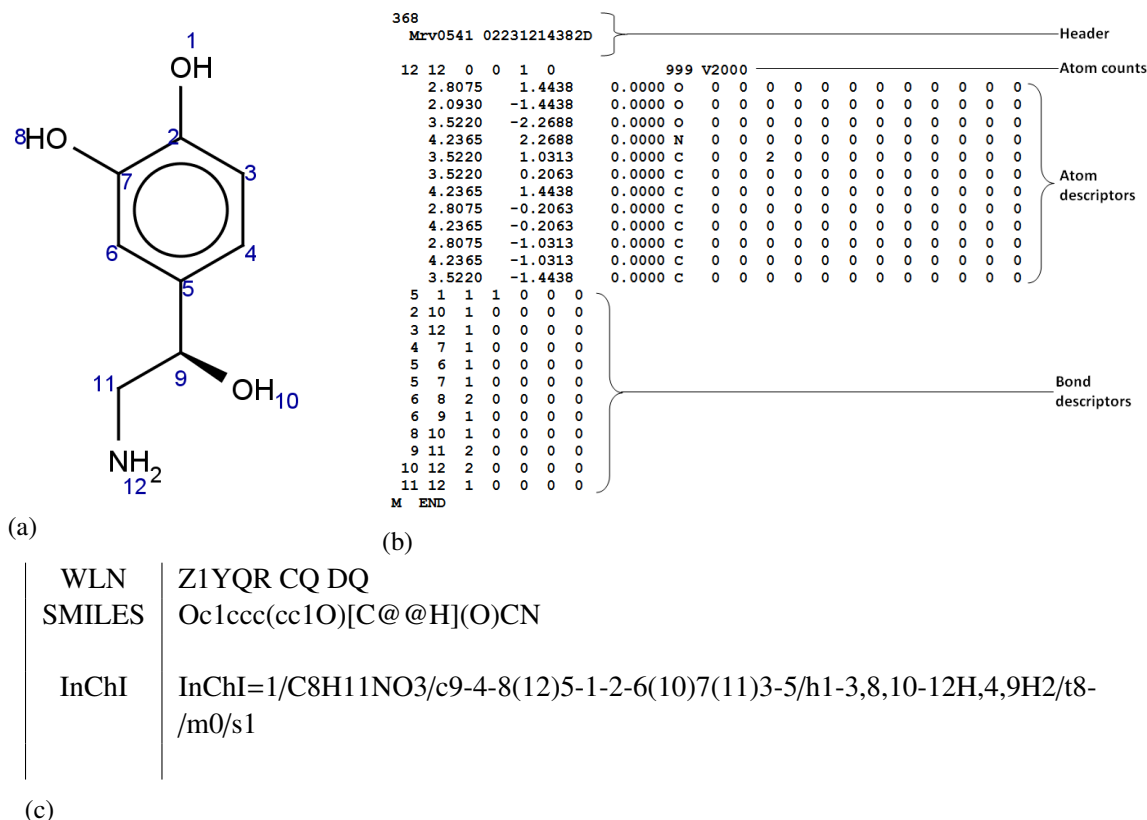


Figure 2.7. Multiple representations of the neurotransmitter noradrenaline, these being a) a graph representation, b) a connection table, and c) linear notations.

counts of features (such as atoms and bonds). The block afterwards gives the details of each atom, including 3-dimensional coordinates and atom type. The final block gives bond information, with details including which atom is connected to what and the order of the bond. An extension to the MOLFile format is the Structure-Data File (SDF), which has become a popular file format as it has the facility for an unlimited number of user-defined properties to be assigned to each atom (Dalby et al., 1992).

2.2.3.2 Other Tabular Notations. It should be noted that, although connection tables are a very useful method for representing molecules due to their (relatively) small storage requirement, there are other tabular forms of representing graphs that are more practical in certain applications. The adjacency matrix (Figure 2.6c) is less storage-efficient but gives all neighbours of a node in one row. This form of representation is often needed in algorithms that rely on matrix operations (examples of which are presented later in the chapter). Even more efficient and compact for storing adjacency information is the adjacency list (Figure 2.6d) which has the smallest time requirements for finding nodes that are adjacent to another. The reader is directed to May and Steinbeck (2014) for further

examples and explanation regarding these three forms of graph representation.

2.2.3.3 Linear Notations. Linear notations of molecules are useful as, like connection tables, they define the exact topology of a molecule with all the necessary atomic and bond features. Unlike connection tables however, they are far more compact and thus have a far smaller storage footprint (usually well under a kilobyte in size for a typical small molecule). Wiswesser Line Notation (WLN) was the first comprehensive linear notation for chemistry, which defined a large and complex set of rules for linear representation of commonly-occurring fragments in molecules (Wiswesser, 1954). This would rely on using single characters to represent fragments, such as the letter "R" for benzene. Due to its complexity however, WLN has since been largely superseded by the Simplified Molecular Input Line Entry Specification (SMILES). SMILES represents a more human-readable notation, which computationally is also easier to calculate (Weininger, 1988). Sybyl Line Notation (SLN) was also another line notation, which provided some alternative features to SMILES, such as a slightly simpler ring definition system (Ash, Cline, Homer, Hurst, & Smith, 1997). SLN though is largely restricted to Tripos applications (*Certara Enhances SYBYL-X Drug Design and Discovery Software Suite*, 2013) and is thus not a widely supported format.

To illustrate the simplicity of SMILES, the SMILES of noradrenaline is represented in Figure 2.7c. Hydrogen atoms are ignored in SMILES unless explicitly defined, and otherwise are inferred from the valences of the non-hydrogen atoms. Atoms are denoted by their symbols, thus carbon is represented by "C" for example. Lower case atoms represent their presence in an aromatic system (like benzene). Brackets denote branches from an atom, such as the (O) representing the oxygen which branches from a carbon atom (atom 9), which is otherwise part of a short chain attached to the benzene ring. A number is used to denote a ring opening, with the same number used to close the ring later in the line. Finally, the @ and / symbols are used to denote chiral bonds.

A somewhat more recent development in linear notation is the *International Chemical Identifier* (InChI). Launched formally in 2005, InChI was developed to give a standardised notation for structural representation. InChI presents different layers of information of a compound separated with forward slashes (Stein, Heller, & Tchekhovski, 2003). Using the example in Figure 2.7c, the InChI version is the first item given (1, in this example), with the molecular formula after the first slash, the connectivity of non-hydrogen and hydrogen atoms after the second and third slashes respectively, then the remaining information relates to stereochemistry. InChIKey is a further development on the application of InChI and internet searching, which compresses InChI into a 27-character hash code to allow for faster searching. Only the first 14 characters are used to describe the topology,

which in most cases is all that is required when performing compound matching (Heller & McNaught, 2009). InChI is used widely as a method for performing quick exact structure matching. The ChEMBL database is one such example which uses InChI and InChI Key for searching (Gaulton et al., 2011). The databases PubChem (Bolton, Wang, Thiessen, & Bryant, 2008) and ChemSpider (Williams, Tkachenko, Golotvin, Kidd, & McCann, 2010) also use InChI as search tools. Although it is quicker to search using InChIKey over InChI, the former has had reported some rare but nevertheless notable collisions (Pletnev et al., 2012). Thus, one should consider double-checking using an alternative notation (such as InChI) when using InChIKey to look up identical molecules (when full accuracy is essential).

2.2.3.4 Canonicalisation. It should be noted that for all of the formats discussed, multiple representations of the same molecule are possible. Consider the SMILES of ethanol - this could be represented as CCO, but also as OCC. Whilst it is possible to attempt to generate all the non-unique forms of a molecule and use these as a "dictionary" when doing structure searching, it is often computationally intractable as this gives rise to a combinatorial problem. *Canonicalisation* methods have thus been developed in order to generate a unique representation of each molecule. Several of these exist, so only a few will be discussed. WLN attempted to enforce canonicalisation via the alphanumeric ordering of fragments. As an example, benzoic acid can be represented as QVR (hydroxyl-carbonyl-benzene), or as RVQ. With alphanumeric ordering however, only QVR is allowed as Q is ordered before R alphabetically (Vollmer, 1983).

Of importance is the Morgan algorithm, which was used to generate unique identifiers for the Chemical Abstracts Service (CAS). Briefly, this algorithm relied on assigning each atom in a structure a unique number based on the connectivity of its neighbours, its neighbours' neighbours and thus for the entire environment about an atom, the atoms with the greater "overall" connectivities being assigned the lowest numbers (Morgan, 1965). CANGEN was developed by Weininger, Weininger, and Weininger (1989) to generate a canonical form of SMILES. This relied on assigning chemical descriptors to atoms as opposed to simply looking at degree sums, then converting the descriptors to integers and ranking them. In order to ensure that each atom had a different number assigned, the ranks were replaced with their corresponding prime numbers. In the next step, each atom was assigned a new rank based on the product of the primes of its neighbours. In cases where ties still existed after the prime products step (often happening in highly symmetric molecules), the ranks were doubled and the lowest rank (selected arbitrarily in ties) was reduced by one. Sadly no formal specification of CANGEN was actually published so a number of different canonicalisation types exist (O'Boyle,

2012). In addition, the CANGEN paper made no mention of how stereochemistry was handled, and used Daylight's aromaticity system. The Chemistry Development Kit (CDK), for example, recently made modifications to the way aromaticity and stereochemistry were handled in their method - two such things which differ in a number of canonicalisation algorithms (May, 2013; O'Boyle, 2012). Work however, is in progress to create a standard canonicalisation specification (O'Boyle, 2012).

The InChI format has a complex yet standardised canonicalisation algorithm which is somewhat similar to the methodology of CANGEN. In brief, atoms are ranked in a structure in order of element type, and then valency. These ranks are then used to represent each atom as its neighbours (in other words, each atom is assigned a descriptor consisting of the ranks of its neighbours), and the atoms are then re-ranked based on the lexicographical ordering of these neighbourhood descriptors. Some additional minor steps are performed to ensure unique ordering, and some additional steps are taken to account for tautomers and isotopes (Hull, Barnard, & Thomas, 2011).

2.2.3.5 Molecular Descriptors. Molecular descriptors represent a diverse range of properties that can be derived from a molecular graph, though do not usually describe a molecular graph in its full detail. These can be categorised into 1, 2 and 3-dimensional (1D, 2D and 3D) descriptors. A 1D descriptor normally is a numeric value representing a property of a molecule on the whole, and thus is rather generalised. Such 1D descriptors can include counts (such as the number of atoms, or number of rotatable bonds) and physicochemical properties, including solubility. 2D descriptors are properties which look at 2D aspects of a molecule, usually on the fragment level, and thus are more specific to a molecule's topology. 3D methods are concerned with the steric space that a molecule can occupy, sometimes referred to as the *conformations* that a molecule can occupy (A. R. Leach & Gillet, 2007b). Here we will focus primarily on 2D methods as these are the descriptors which are most concerned with hyperstructures.

2.2.3.6 Molecular Keysets and Fingerprints. *Molecular keysets*, sometimes referred to as fragment codes, are dictionaries of pre-defined fragments. Such dictionaries can be applied to search a database of compounds, given that the fragments, or *features* of the database compounds have already been defined. Traditionally the presence or absence of a feature in a molecule has been represented using a *bitstring*. With an example, assume that a keyset is made of 100 fragments. This would likewise correspond to 100 respective bits in a query molecule, all initially set as 0 (or "off"). Upon identifying a fragment in the molecule from the dictionary, the respective bit would be set to 1 (or "on") in the query molecule's bitstring. Molecular keysets were initially developed for use with punched card sorters or optical coincidence cards (Willett, 1987), the original rationale to improve the efficiency

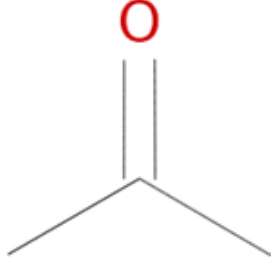
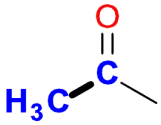
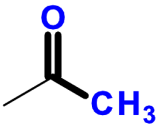
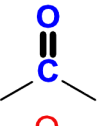
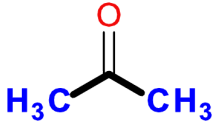
Molecule	Atom pair	hash-code	Example
	C.X3 -2 -CX1	4603c2fa	
	O.X1 -3 -CX1	3a03d3c0	
	O.X1 -2 -C.X3	de658b6a	
	CX1 -3 -CX1	1fcf3d5d	

Table 2.1

An example of the atom pairs method describing propanone. Atoms are represented by their atom type, Pi electrons (represented as a dot), and number of non-hydrogen connections (X, then a number). The hyphens delimit the atom descriptors and the path distance. Hash-codes were generated using a cyclic redundancy check hashing algorithm (*crc32*).

of structure and substructure searching (Christie, Leland, & Nourse, 1993; Hagadone, 1992). More recently they have been used for virtual screening methods (M. Johnson, Basak, & Maggiora, 1988; Krasowski, Siam, Iyer, & Ekins, 2009; Papadatos et al., 2009). The 960 keys released by MDL are one example of a keyset, which were originally constructed to optimise substructure searching (Durant, Leland, Henry, & Nourse, 2002).

Molecular keysets represent a poor method for uniquely defining molecules because of the ambiguity of molecular representation. Several molecules may share the same collection of fragments, yet the molecules differ as the fragments are arranged in different topologies. Furthermore, it is awkward to update a pre-coded fragment dictionary for a large database of molecules, for the keyset must be recalculated for each molecule. These problems, however, are unsurprising given that keysets were not designed to fully describe a molecule's structure in the first place. Thus, the application of more rigorous notations like linear notations and connection tables is more suitable for defining structures (Willett, 1987). The ambiguity of keysets, however, makes them far better suited to similarity and substructure searches, which will be explained in more detail later in this chapter.

An extension from molecular keysets are *hashed fingerprints*, which describe a molecule with a series of bits, but following an algorithmic generation of bits from a rule set, as opposed to using a pre-defined dictionary of fragments. This can give an enormous variety of bits, and thus a far

more diverse representation of compounds. Crucial to the development of hashed fingerprints is the generation of fragments. Armitage and Lynch (1967) described one such method, where molecules were abstracted first as atom-bond-atom pairs. These pairs were then linked to yield a series of chains of atoms and bonds, which were common between two molecules. The method was ultimately used to yield a non-redundant set of fragments, which could be used for describing molecules and performing basic similarity calculations. Crowe, Lynch, and Town (1970) extended this work by describing two types of fragments referred to as "augmented atoms" and "bonded pairs." The augmented atoms in this case referred to taking an atom and encoding its immediate neighbours and bond types attached to it, yielding a small fragment of a molecule. Bonded pairs encoded the information around a bond, being the bond order, the two atom types on the bond, and the types of bonds attached to these atoms (though not the atom types of their neighbours). A frequency analysis of the fragments generated from a 5% sample of the CAS database was performed, which at the time contained almost 600 000 known synthesised compounds (Crowe et al., 1970). Adamson et al. (1973) encoded the aforementioned fragments into bitstrings, which were used as part of a presence-or-absence of feature screening process. It should be noted that sometimes the term "fingerprint" refers simply to hashed fingerprints, but in other publications the term may be used to refer to any method that generates a bitstring from a molecule (Owen, Nabney, Medina-Franco, & López-Vallejo, 2011; Willett, 2009).

The *atom pairs* method described by Carhart, Smith, and Venkataraghavan (1985) was proposed as a descriptor for Structure-Activity Relationship (SAR) analysis. The atom pairs method started by selecting two atoms and describing them by their element type, the number of Pi electrons, and how many non-hydrogen connections they had. The number of atoms between the two atoms (including the atoms themselves) was also included as a descriptor. Hashing could then be performed to turn the generated descriptors of a molecule into integers. This generated a simple yet descriptive fingerprint for a molecule (Table 2.1).

Several other fingerprints have been developed since the atom pairs method. A selection of these will be overviewed here, though the reader is directed to a review by Cereto-Massagué et al. (2015) for more information on fingerprint types. Daylight introduced path fingerprints, a fingerprint similar to the chain-building method of Armitage and Lynch (1967) which encoded descriptors of atoms along paths of set lengths in a molecule. As an example, if a length of 4 was chosen then paths of 1, 2, 3 and 4 atoms were enumerated in a molecule (James, Weininger, & Delany, 2004). *Extended connectivity* fingerprints, also known as radial or circular fingerprints, described a molecule via fragments instead of paths, exploring the neighbourhood of an atom in a radial manner, similar to the way the Morgan

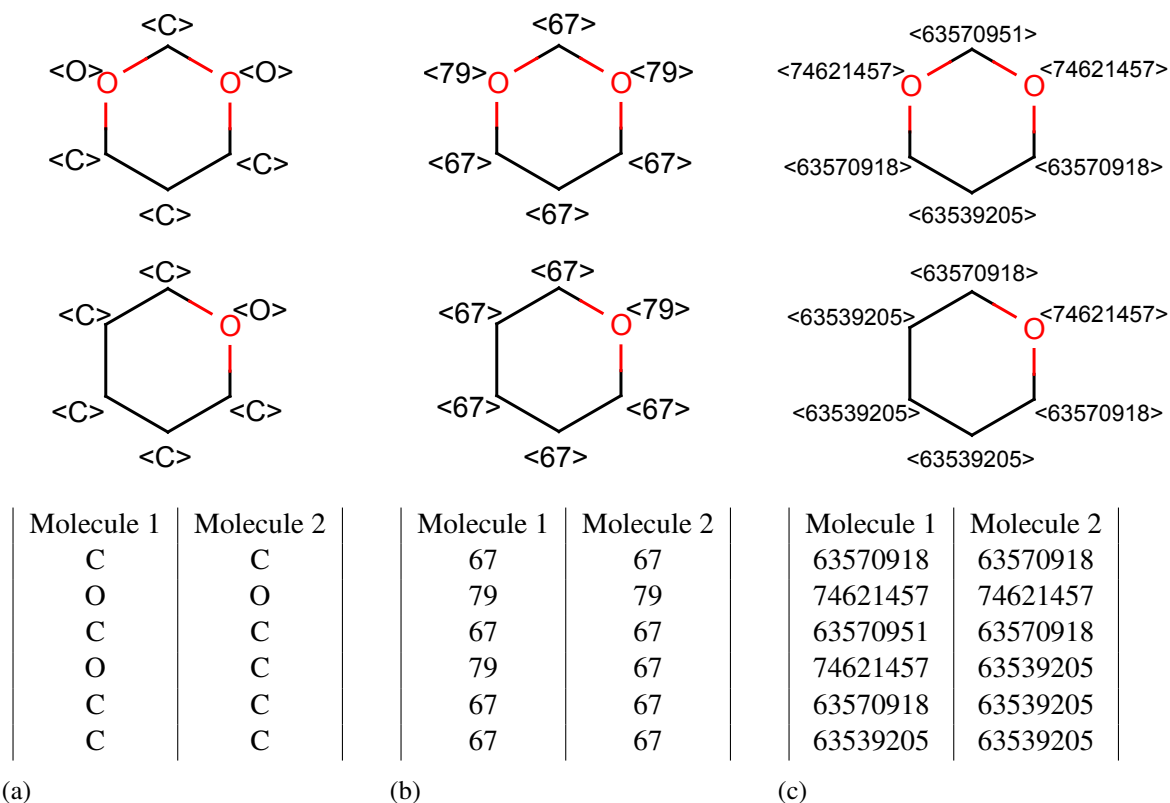


Figure 2.8. An example of how extended connectivity fingerprints are computed for two molecules - molecule 1 (top) and molecule 2 (bottom). a) Assuming a simple atomic descriptor of element type, descriptors are first assigned to each atom. b) These descriptors are then hashed to give integers. c) Neighbours of each atom are then taken into account. In this case, the neighbour integers are appended to the atom in question and the enlarged descriptor is re-hashed into a different integer. Only one neighbourhood iteration is shown but any number of iterations is possible.

algorithm worked. These were originally developed by Dubois (1973), who introduced this fingerprinting method to generate a series of fragments for compound screening. These were not however referred to as fingerprints, instead as *Fragments Reduced to an Environment that is Limited* (FRELs) (Dubois, Panaye, & Attias, 1987). Scitegic (now owned by Accelrys/BIOVIA and Dassault) however used the method to generate fingerprints, starting with a variant of the Daylight atomic invariants rules (Weininger et al., 1989) as a descriptor (Rogers & Hahn, 2010). Figure 2.8 gives an example of how extended connectivity fingerprints are calculated.

2.2.3.7 Reduced Chemical Graphs. Reduced Chemical Graphs are loosely related to graph-theoretical hypergraphs - graphs where an edge can represent several vertices in the graph that the hypergraph has been abstracted from (Bunke, Dickinson, Kraetzl, Neuhaus, & Stettler, 2008). In reduced graphs however, a vertex represents one or more atoms (and zero or more bonds) from a molecule. Several forms of abstraction exist, where reduced graph nodes can represent ring systems

and ring linkers, or functional groups (Gillet et al., 1987). Reduced graph similarity can be more useful than conventional graph similarity for virtual screening, when one wishes to seek less apparent structural similarities. This is particularly important in scaffold hopping, where relying on exact topological similarity may fail. Reduced graphs have been used outside of similarity searching, including in clustering and Markush structure searching (Birchall & Gillet, 2010).

2.3 Substructure Searching

Modern cheminformatics search techniques can be divided into three classes: exact; substructure; and similarity searches. Exact searches test for the identity of one molecular graph with another, which essentially translates to the labelled and weighted graphs being isomorphic, possessing the same topology of atom and bond labels. Exact matching is now relatively trivial with the ability to quickly determine isomorphism, in addition to standardised string matching, such as the matching of InChI strings. Substructure searching tests for subgraph isomorphism, finding the presence of a graph within another graph. Similarity searching by contrast attempts to produce a numerical measure which represents a degree of shared features between two molecules (discussed in detail in Chapter 3).

The main use of substructure searching is the finding of analogues of a particular scaffold, which is particularly useful when most similarity methods (which consider the molecule on the whole, as opposed to the local level) would fail to exhaustively find such analogues (this however is not the case when using asymmetric coefficients (Willett, 2009), though these will not be discussed here). An example substructure search with noradrenaline is shown in Figure 2.9. A-77636 for example, possesses a similar level of activity to noradrenaline, against the alpha-2a adrenergic receptor.

Substructure searching commonly relies on *subgraph isomorphism* - testing to see whether a subgraph within the database molecule(s) exists, which is isomorphic to the query. In chemistry it is somewhat simpler to perform graph matching than in most other graph theory contexts, due to the comparatively low connectivity (or in other words, the degree of each node is generally small) of chemical graphs, and that molecular graphs are generally labelled and weighted. The latter property in particular can be used to specifically tailor graph matching algorithms to hasten the search for subgraphs, for labelled nodes are usually assigned to nodes of the same type in the other graph (and the same being for bonds).

In an ideal case, graph matching algorithms are used to perform substructure searching. In practice

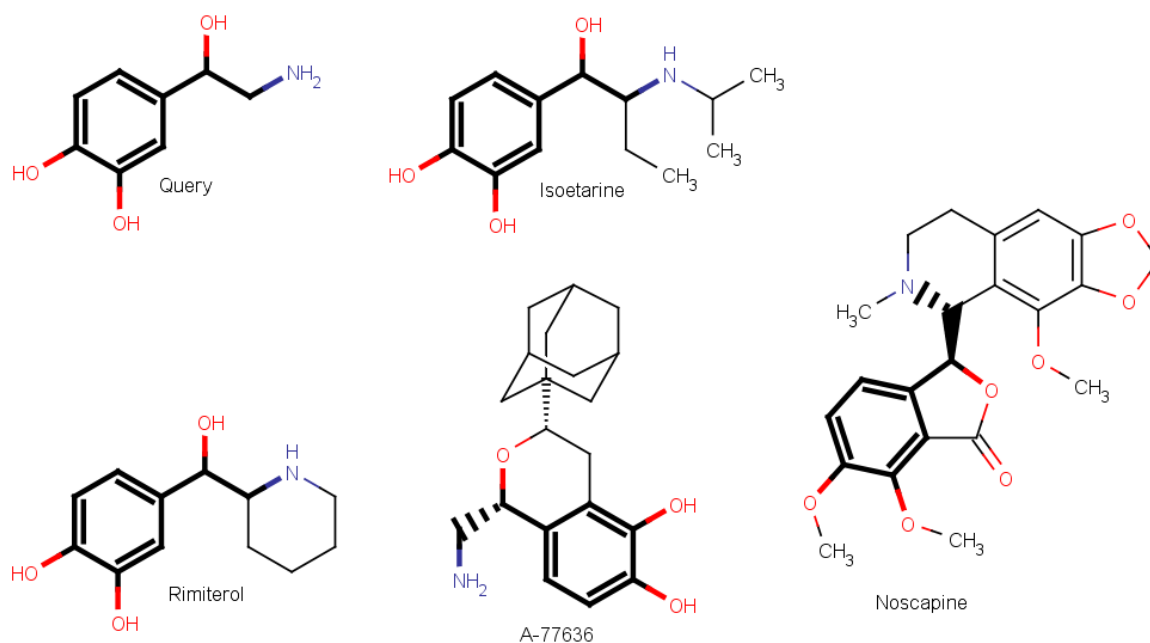


Figure 2.9. A small number of hits from a substructure search using noradrenaline as the query, in the ChEMBL14 database.

however, graph matching algorithms are somewhat slow and thus impractical for large databases. Molecular keysets and/or fingerprints can be used to act as screens - molecules in the databases are selected which possess all 'on' bits that the query has, and then these molecules can be subjected to the computationally intensive graph-matching algorithm to complete the substructure search. In most cases involving substructure searching, the user is only interested ultimately in the presence or absence of a given substructure rather than partial matches. This presence/absence requirement is advantageous for improving the speed of substructure searching, as for a given query, the rarest features can be prioritised for in searching (e.g. an arsenic atom is rarer than a carbon atom, so in all database molecules the arsenic would be the first feature to look for). This was demonstrated by Ehrlich and Rarey (2012), who showed that such prioritisation could dramatically improve search speeds for substructure searching and make it easily feasible for searching of large datasets.

2.3.1 Subgraph Isomorphism Algorithms

Subgraph isomorphism algorithms will be briefly described here, primarily because a number of MCS algorithms (in particular, subgraph enumeration algorithms) rely on subgraph isomorphism algorithms to calculate both the cMCS and dMCS (Section 2.2.2.2). This of course means that one can also view an MCS as a series of common subgraph isomorphisms between two graphs.

2.3.1.1 Ullmann's Algorithm. Ullmann's algorithm is a backtracking algorithm (McGregor, 1982) that was developed to enumerate all subgraph isomorphisms between two graphs. It was widely used at the time, and is often used as a benchmark for time-comparison of graph-matching algorithms (Cordella, Foggia, Sansone, & Vento, 1999; Ehrlich & Rarey, 2012). The algorithm's aim was to create all the match matrices which denote the subgraph isomorphisms from G_2 onto G_1 . A match matrix has $m \times n$ rows by columns, where m is the number of nodes from G_1 and n is the number of nodes from G_2 . For a match matrix that denotes a unique subgraph isomorphism, each column is a set of 0s with one of the elements of the column set to 1 - the 1 indicating that the relevant node from G_2 matches G_1 . The algorithm would start by constructing an adjacency matrix for each of the two graphs being matched. An initial match matrix would be computed based on degree correspondences of atoms (a 1 would be set for a pair of atoms if the degree in G_2 was greater or equal to the atom in G_1). As the initial matrix would have an "excess" of elements set as 1, the algorithm would enumerate every valid form of the matrix, each form differing by at least one of these elements (which would be set as 0). In order to verify that a match matrix specified a subgraph isomorphism from G_2 to G_1 , a matrix C would be constructed such that $C = M^T(MB)^T$, where M is a match matrix and B is the adjacency matrix of G_2 . If all of the 1-elements in the adjacency matrix of G_1 were also 1 in C , then a subgraph isomorphism was identified. Ullmann also described a simple refinement procedure which set 1 elements in the initial match matrix to 0 based on the absence of a neighbour in the match matrix (to the vertex in both graphs) which is also set to 1 in the match matrix (Ullmann, 1976). As Ullmann's algorithm enumerates all possible subgraph isomorphisms, it is somewhat unfeasible for finding the MCS. However, it has been shown to be very effective for simply proving that a subgraph isomorphism exists between two graphs (Ehrlich & Rarey, 2012).

Gouda and Hassaan (2012) applied two extensions to the Ullmann algorithm to get a boost in speed of between 1 and 3 orders of magnitude. The first modification involved sorting the input vertices in order of connectivity, where the first vertex in the list to be analysed was that of maximum degree in the graph, whereas subsequent vertices were those which have the highest connectivity to previous members in the list (in descending order). The second modification relied on the fact that a vertex in the subgraph will have a neighbourhood of vertices and edges which are a subset of the vertices and edges of the equivalent vertex in the parent graph. Thus, the labels of the vertices and edges will also be identical. To exploit this, they constructed a bit matrix where the dimensions were the distinct neighbourhoods of the 2 graphs being compared, a 1 in the matrix signifying a neighbourhood match, and a 0 otherwise signifying no match. This bit matrix was thus used to perform vertex matching

between the graphs, but at a significantly reduced search space.

2.3.1.2 VF2. The VF2 algorithm (Cordella, Foggia, Sansone, & Vento, 2001) was originally developed to test for subgraph isomorphism in directed graphs. The VF2 algorithm attempts to enumerate subgraphs, given an input matching to start with (that is, anything other than 2-vertex subgraphs). VF2 is a backtracking depth-first search, as opposed to a breadth-first search. It also introduced the concept of a *state* - a subset of nodes (and edges) of the common subgraph thus identified. Although originally designed to be used on directed graphs, the algorithm has been implemented in the CDK for use with molecules to find the cMCS (Rahman, Bashton, Holliday, Schrader, & Thornton, 2009). It applied heuristics which relied on matching nodes and edges in the states to find subgraph isomorphisms quickly, based on the topology of the immediate neighbours of a given state. VF2 alone has been shown to be significantly faster than Ullmann's algorithm for substructure searching, particularly when the aim is to find only one subgraph isomorphism, as opposed to all (Ehrlich & Rarey, 2012).

2.4 Finding the Maximum Common Subgraph

Of particular relevance to this thesis is the MCS, for the MCS is often used in the construction (and application) of hyperstructures in order to achieve the optimal overlap and compression. Aside from hyperstructures, uses for the MCS include bioactivity prediction of compounds (Y. Cao, Jiang, & Girke, 2008), protein function prediction (Borgwardt et al., 2005), similarity searching (Raymond & Willett, 2002a), and in malware detection (Sirageldin, Selamat, & Ibrahim, 2011). It should be noted that in chemistry, the MCES is generally more useful than the MCIS as the MCES covers a greater proportion of the two chemical graphs than the MCIS does. In a hyperstructure context for example, the MCES would be expected to give a superior compression effect. The MCIS however may be more suitable for reduced graphs, where nodes in said reduced graphs encompass many features of molecules. Also, the MCES can be unsuitable in reduced graphs due to tendency for there to be no compatible edges between more complex reduced graph methodologies (Barker et al., 2006). This of course highlights a use for the MCIS in identifying more sparse similarities that the MCES would fail to identify between two graphs.

Determination of the MCS is an example of a Non-Deterministic polynomial time Complete problem (NP-complete). NP problems are decision problems (i.e. there exists a yes/no answer) such that a solution is (currently) impossible to find in polynomial time, although given an "oracle" or a previously found solution, the solution is verifiable in polynomial time. NP-complete problems are

the hardest of all NP problems to solve, where an NP-complete problem can be reduced to another NP problem in polynomial time. This has important implications, for if a polynomial solution can be found for one NP-complete problem, then all other NP-complete problems can be transformed to become solvable in polynomial time too. It should be noted that a related class of problems exists called NP-hard, which are at least as difficult to solve as NP-complete. These however do not necessarily have to belong in NP, and may not even be decision problems (although many do belong in NP). An NP-complete problem can also be transformed into an NP-hard problem in polynomial time. (Cook, 1971; Garey & Johnson, 1979). It should be noted that it is currently unknown whether $P = NP$ or $P \neq NP$ (P representing problems solvable in polynomial time). There is a lot of interest in solving this, so much so that a million dollar cash prize will be awarded to the group who finds the answer to this question (Carlson, Jaffe, & Wiles, 2006). Many NP-complete problems that exist are currently tackled using approximate solutions in order to get a near solution, rather than obtaining the exact solution. Other NP-complete problems mentioned in this chapter include the maximum clique, minimum vertex cover and maximum independent set problems.

As mentioned previously, the maximum clique problem can be reduced to finding the minimum vertex cover (or maximum independent set) in a complement graph (Bomze et al., 1999). Although we will not review said algorithms here, interested readers are directed to a paper by X. Huang, Lai, and Jennings (2006), where a number of MCS-finding approaches are reviewed, in addition to attempts to use minimum vertex cover algorithms to obtain the MCS. In addition, two promising approximate algorithms by (Pullan, 2009) and (Ugurlu, 2012) are fast algorithms which often yield optimum solutions for the minimum vertex cover problem, which can otherwise be used to yield sharp lower bounds for maximum clique algorithms.

2.4.1 Application of Cliques to Find the MCS

The idea of applying clique detection to find the MCS (specifically the MCES) of two graphs seems unintuitive at first, particularly as the graphs themselves may not exhibit cliques. The key to this lies in the *modular product* of two graphs, where maximum clique detection can be used to find the MCIS. Maximum clique detection is an NP-hard problem, having been shown that it is hard to approximate in polynomial time within a factor of $n^{1-\epsilon}$ for n nodes in a graph, and $\epsilon = k \log n$, where k is a constant and $\epsilon > 0$ (Håstad, 1999). As will be shown later, several algorithms exist in the literature which have been developed for clique detection.

The modular product of two graphs, sometimes referred to as a compatibility graph, is a graph with

the vertex set $V_1 \times V_2$ of graphs G_1 and G_2 respectively. If the edge set of G_1 is $E(G_1)$, where an edge is a pair of vertices (e.g. (u_1, u_2)), and by analogy for $E(G_2)$ and G_2 , then an edge between two vertices exists in the modular product where

$$(u_i, u_j) \in E(G_1) \text{ and } (v_i, v_j) \in E(G_2) \text{ or}$$

$$(u_i, u_j) \notin E(G_1) \text{ and } (v_i, v_j) \notin E(G_2)$$

with (u_i, u_j) being a pair of vertices in an edge in G_1 , and (v_i, v_j) being a pair of vertices in an edge in G_2 . On construction, the maximum clique(s) in the modular product correspond to the MCIS between the two graphs (Barrow & Burstall, 1976; Kozen, 1978). In weighted and labelled graphs, an additional condition applies such that only nodes of the same label are matched, and likewise for the edge-matching conditions, i.e:

$$w(u_i, u_j) = w(v_i, v_j)$$

where $w()$ gives the vertex and edge labels for a pair of vertices in an edge.

Using Figure 2.10 to demonstrate the modular product, it can be observed that edges exist between nodes of the modular product when an edge exists for G_1 for a given two vertices, and an edge also exists for G_2 . Vertices A and B in G_1 and X and Y in G_2 show this, for an edge exists between both A and B in G_2 , and X and Y in G_1 , thus giving rise to an edge between vertices 1 and 5. An edge also exists between vertices 1 and 12, for no edge exists between the corresponding G_2 vertices A and D, and neither is there an edge between X and Z in G_1 . No edge however exists between vertices 1 and 2, for whilst there exists an edge between X and Y in G_1 , vertex A is not joined to vertex A in G_2 . It should also be noted that four cliques (and thus 2 unique MCISs) are highlighted in the example. Clique (4,8,12) for example corresponds to the MCIS involving vertices (B,C,D) in G_1 , and (X,Y,Z) in G_2 .

2.4.1.1 Adaptations to find the MCES. Though the method discussed so far finds the MCIS, it is generally the MCES that is sought in chemoinformatics (the two are contrasted with their modular products in Figure 2.12). Notably, finding the MCES is computationally simpler than the MCIS as one must take into account both vertex and edge labels, rather than just vertex labels as in the MCIS (and thus the modular product will contain fewer nodes and edges). Whitney (1932) proved that

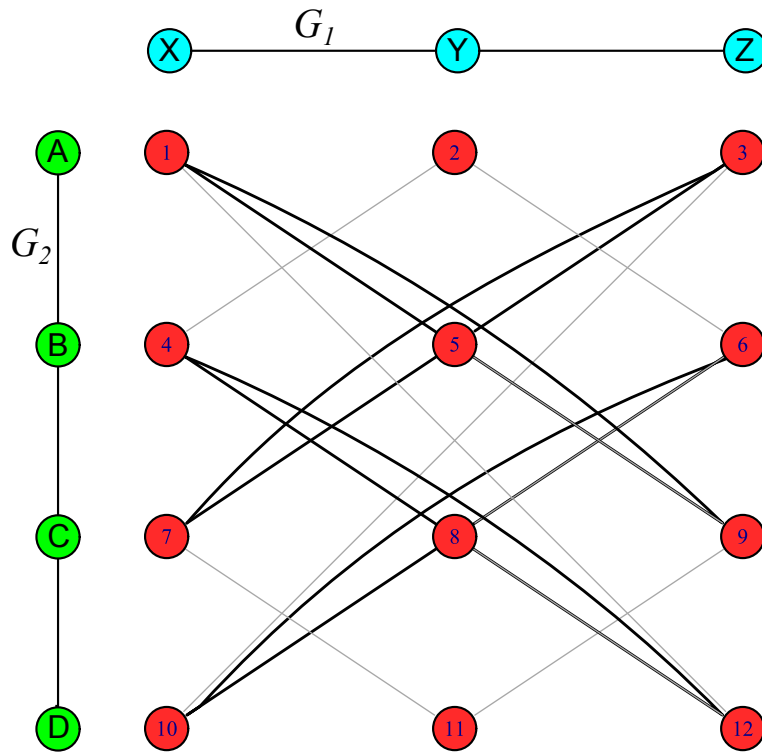


Figure 2.10. Modular product of two graphs G_1 and G_2 , where G_1 is a linear connection of 3 nodes (X, Y and Z) and G_2 a linear connection of 4 nodes (A, B, C and D). Edges in bold are part of the 4 cliques that exist, whereas edges in light grey are not part of these cliques. The numbers have been assigned to each node of the modular product for referencing purposes.

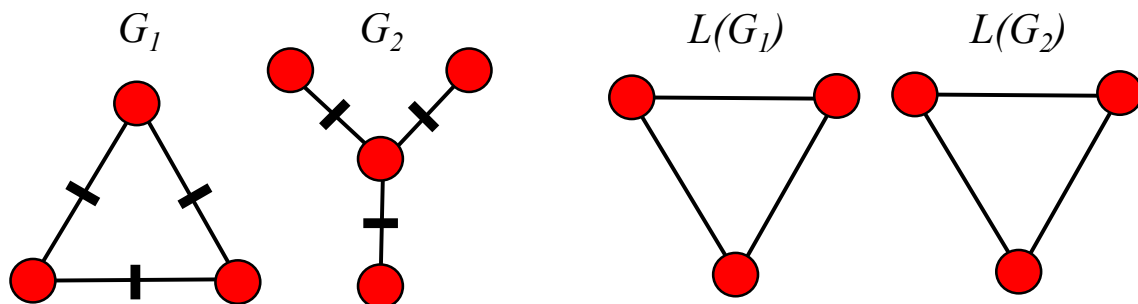


Figure 2.11. An example of the Δ -Y exchange phenomenon adapted from Raymond and Willett (2002b).

it is possible to reconstruct a graph from its respective line graph, and *vice versa*. By analogy, it is therefore possible to construct the MCES from the MCIS. When seeking the MCES, a modular product should be constructed *from the line graphs instead* of the original graphs. Thus, a vertex in the modular product will now represent an edge match between the two original graphs, instead of a vertex match. Of course, one may instead construct the modular product directly from the original graphs, but ensuring that a vertex in the modular product represents an edge match (while also ensuring that the vertex labels on each edge also correspond). Further details of this methodology are

given in Chapter 6, in section 6.2.2.

Reconstruction of line graphs back to the original graphs is possible in all cases with one exception - the Δ -Y exchange, where the conversion of the two graphs in Figure 2.11 (the delta and the Y-shaped graphs, hence the name Δ -Y) leads to two edge graphs which are isomorphic, as opposed to unique. The Δ -Y exchange thus leads to erroneous results for the MCES when translating back from line graphs to the original graphs, though such erroneous results are easy to prune - for the presence of Δ -Y exchanges leads to results with different degree sequences (the MCES in 2 graphs should have the same degree sequences). This line graph exploitation was used in the work of Koch (2001) who referred to the modular product as the "edge product" when seeking the MCES. This technique was also used in the MCS-finding algorithms TOPSIM (Durand, Pasari, Baker, & Tsai, 1999) and RASCAL (Raymond, Gardiner, & Willett, 2002b), the latter of which was developed here at Sheffield.

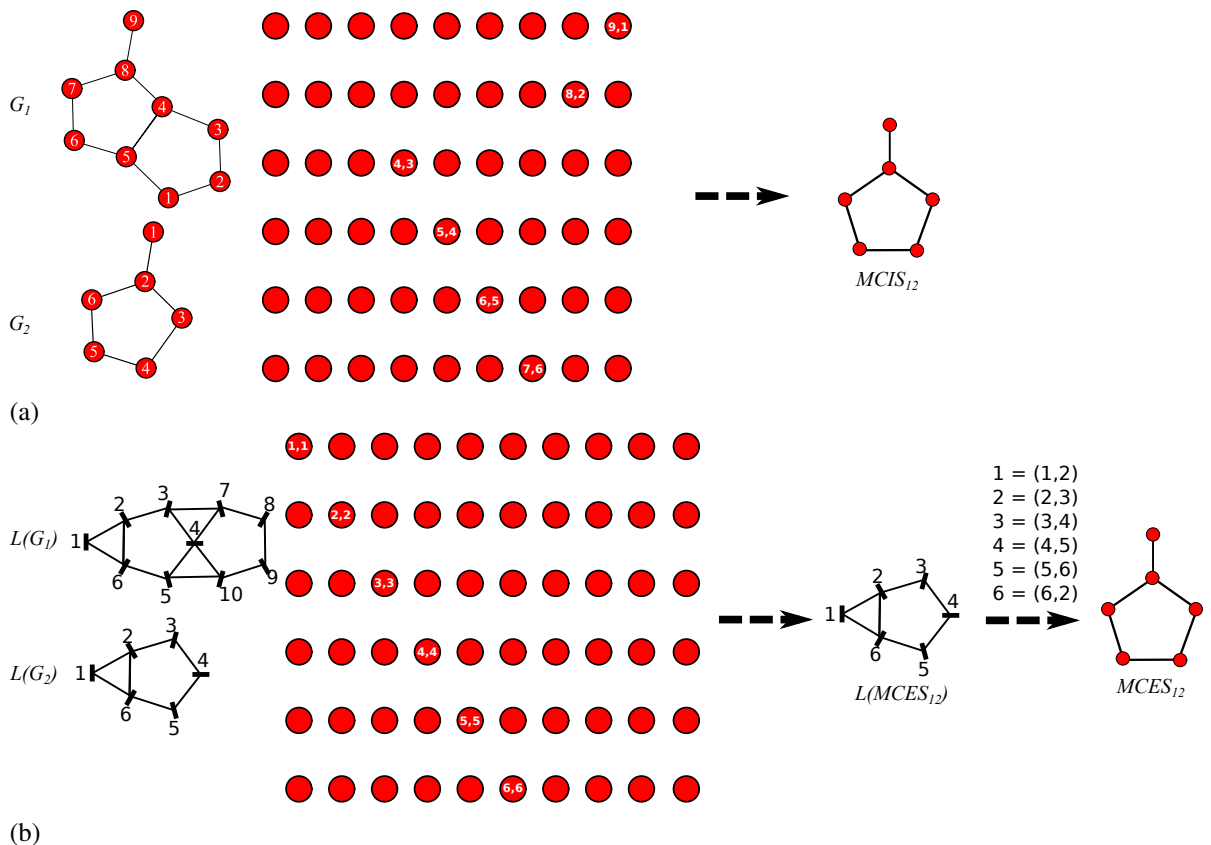


Figure 2.12. Finding the 2.12a MCIS and the 2.12b MCES of two graphs, using the modular products of G_1 and G_2 . The nodes of the modular product cliques used are labelled (edges not shown) in both instances. Note that the MCIS and MCES in this example are isomorphic, but as shown in Figure 2.5 that is not always the case.

2.4.2 MCS Categories - Modifying the Modular Product

As one may have gathered, the modular product graph constructed from two molecular graphs can be large, and more importantly, highly dense in edges. A large edge density often results from the sparseness in degree of chemical graphs, leading to a more time-consuming search process for clique detection algorithms. Although a more powerful clique detection algorithm can be used, it is also worth simplifying the modular product without removing chemical meaning (or size) from the MCS retrieved.

Kawabata (2011) introduced the *topologically-constrained dMCS* (tdMCS), a concept which has parallels in older publications, though is not defined as tdMCS in those publications (Barker et al., 2006; Klinger & Austin, 2005; Takahashi, Sukekawa, & Sasaki, 1992). The tdMCS has the attractive property of producing a dMCS that yields more realistic alignments, whilst being faster to calculate than a conventional dMCS (i.e. if a geometric alignment is obtained using an MCS in a similar fashion to Kawabata (2011), the application of a strict tdMCS will yield a non-distorted geometric alignment). Although the dMCS is suitable for mapping multiple “fragments” between two molecules, no constraints on the distance between the fragments are considered. Consider the example in Figure 2.13a, where two sets of edges are mapped which when, for instance, aligned in 3D or used to produce a hyperstructure, would otherwise yield nonsensical results.

Construction of the tdMCS relies on the concept of topological distance $T(a, b)$ between two atoms (or edges) a and b . In this case, topological distance refers to the minimum path distance between two features in the graph. The tdMCS is the dMCS resulting from a constraint applied when constructing edges in the modular product graph:

$$|T_A(a_i, b_i) - T_B(a_j, b_j)| \leq \theta \text{ for } 1 \leq i < j \leq |m|$$

for two graphs A and B . θ represents a threshold on the maximum allowed difference in distance between two pairs of atoms (or edges). $|m|$ represents the number of nodes in the modular product graph. Essentially, any edge in the modular product that violates this constraint is deleted, the maximal clique(s) of which correspond to tdMCSs. Figure 2.13b is a tdMCS where $\theta = 0$. Of note, the same constraint can be applied to the cMCS to yield the tcMCS, in analogy with the tdMCS.

Kawabata (2011) remarked that the tdMCIS yielded better agreements with alignments of superimposed ligands in ligand-protein structures, than with the dMCIS. In addition, Klinger and Austin

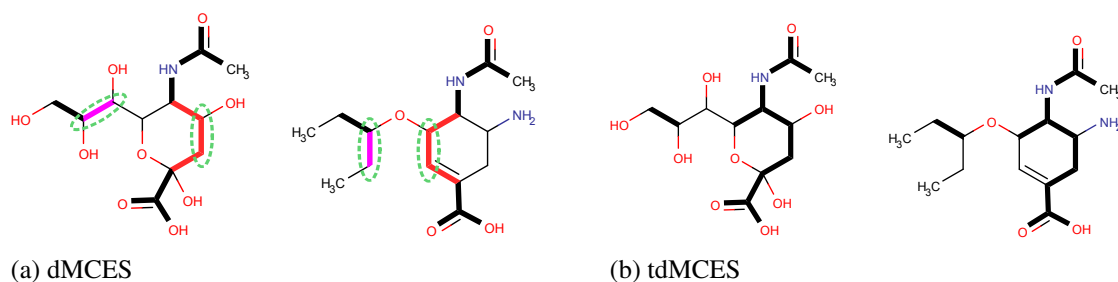


Figure 2.13. An example of two molecules where the dMCES yields an unsatisfactory alignment. The two sets of offending edges are coloured in purple and scarlet. In the dMCES example, a pair of corresponding edges are circled. In the first molecule the topological distance between the edges is 3, whereas in the second molecule the distance is 2. These edges are thus matched differently in the tdMCES to satisfy the distance constraint of 0. Adapted from (Kawabata, 2011)

(2005) tested a neural network MCS algorithm on two types of MCS, referred to as “bond types” and “topological distances.” The former was a concept similar to the dMCIS whereas the latter would be a tdMCIS with a threshold difference of 0 bonds. The tdMCIS in this study was shown to generally obtain superior recall statistics than the dMCIS in virtual screening of a 10102 compound dataset.

A complex set of heuristics was introduced by Raymond, Gardiner, and Willett (2002a), in attempt to retrieve an MCS equivalent or near in size to the dMCS whilst significantly reducing the required search time. The heuristics involved deleting both nodes and edges from the modular product graph. Node deletion heuristics relied on local symmetry in benzene rings and/or aromatic rings with common substituents to remove redundant mappings. Edge deletion heuristics relied on exploiting features in ring systems, as well as certain substructures containing large numbers of Carbon-Carbon single bonds.

2.4.3 Clique Detection with Graph Colouring

Partly as a result of the application of cliques to obtain the MCS of two graphs, there have been a number of algorithms developed to allow the detection of cliques. One class of algorithms relies on the principle of graph colouring, an NP-complete problem which stemmed from the four colour problem, a theorem which was initially proposed by Francis Guthrie in 1852. This theorem was eventually proved (computationally) by Appel and Haken (Appel & Haken, 1977). Originally colouring was proposed as a method for colouring countries on a map (i.e. no neighbouring country would have the same colour). Mathematically, graph colouring (of vertices) can be represented as a function $c(v)$ for a given vertex v , such that $c(v_1) \neq c(v_2)$ when $(v_1, v_2) \in E(G)$. As with maps, each vertex must have a colour that is different from its neighbours' colours. The smallest number of unique colours in

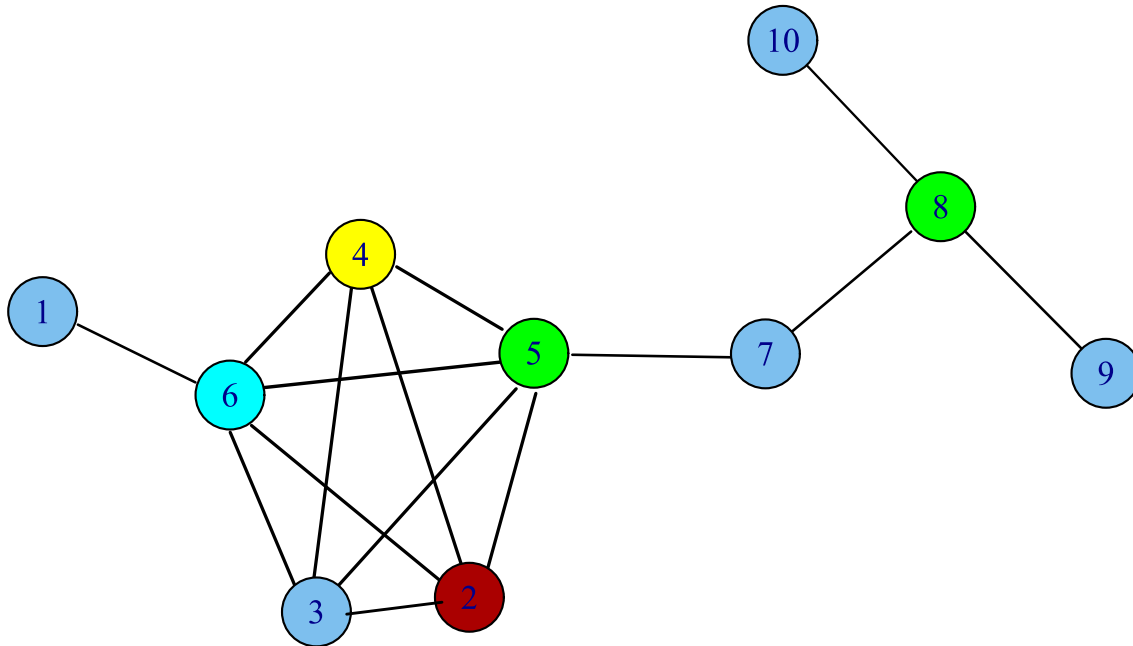


Figure 2.14. A demonstration of a coloured graph which contains a clique (where the clique vertices are $\{2, 3, 4, 5, 6\}$). The chromatic number here is 5.

a graph is denoted by its *chromatic number*, sometimes written as $\chi(G)$ for a graph G (Diestel, 2006).

On inspection of Figure 2.14 it soon becomes clear how graph colouring is useful in clique detection, for all the nodes in a clique have different colours. Thus, a clique can be easily detected by merely finding that for a given node, all of its neighbours possess different colours. It is ultimately this fact that a number of clique detection algorithms rely on, despite graph colouring also being NP-complete.

2.4.4 Clique Detection Algorithms and the MCS

In its simplest form, a (backtracking) clique detection algorithm involves traversing a search tree based on a recursive definition (Pardalos & Xue, 1994; Raymond, 2002)

$$\omega(G) = \max\{1 + \omega(N_G(v)), \omega(G \setminus v)\}$$

where $\omega(G)$ is the maximum clique of graph G (usually the modular product in this work), $N_G(v)$ is the set of vertices adjacent to vertex v in G , and $G \setminus v$ is graph G with vertex v deleted. In pseudocode the algorithm can be represented as in Algorithm 1. A high-performance algorithm based on this methodology is the Carraghan-Pardalos algorithm (Carraghan & Pardalos, 1990), which follows a non-recursive form of Algorithm 1. Here, the vertices in G are in ascending order of degree, where

v_1 has the smallest degree in G , v_2 has the smallest degree in $G \setminus v_1$, and v_k is the vertex of smallest degree in $G \setminus \{v_1, \dots, v_{k-1}\}$.

Data: vertex set G (of modular product), current clique C , max clique $\omega(G)$

Result: Maximum Clique $\omega(G)$

Function FindCliques(G):

```

while  $G$  is not empty do
  select  $v$  from  $G$ ;
   $C = C \cup v$ ;
  FindCliques( $N_G(v)$ );
  if  $|C| > |\omega(G)|$  then
    |  $\omega(G) = C$ 
  end
   $G = G \setminus v$ ;
end

```

Algorithm 1: A simple backtracking maximum clique detection algorithm

Readers interested in finding more about clique detection algorithms are directed to a review by Gardiner, Artymiuk, and Willett (1997), as well as a review on some alternative applications on clique detection (Pardalos & Rebennack, 2010). A number of clique detection algorithms will now be discussed.

2.4.4.1 The Bron-Kerbosch Algorithm. One of the most widely-used maximal clique detection algorithms is the Bron-Kerbosch algorithm (Bron & Kerbosch, 1973). In its simplest form, the algorithm can be represented recursively as in Algorithm 2. The algorithm is in many ways similar to the basic maximum clique detection algorithm, in that it recurses through neighbourhoods of selected vertices in the graph. However, a notable feature is that it keeps track of previously-used vertices (represented as X in Algorithm 2) to avoid futile backtracking. In addition, it reports all maximal cliques found, not just a maximum. A modification of the algorithm by the same authors, and more formally defined by Cazals and Karande (2008), features the utilisation of a “pivot vertex” selected from $P \cup X$. Vertices would only be considered for clique exploration if they were not adjacent to the pivot vertex.

Koch (2001) developed three algorithms which extended the Bron-Kerbosch algorithm, all of which relied on removing redundant cliques that did not contribute to finding the MCS when using the modular (or edge) product of two graphs. The principle behind this was the application of c -edges. Given a modular product of a graph with itself, the c -edges are the edges in the product where the edges (e_1, f_1) and (e_2, f_2) possess a common vertex, where (e_1, e_2) is an edge in G_1 and (f_1, f_2) is an edge in G_2 . In this case for clique detection, G_1 is isomorphic to G_2 . A clique is spanned by

Data: vertices in clique R , current vertex set (of modular product graph) P , previously-processed vertices X

Result: Maximal Cliques C

Function BK(R, P, X):

```

if  $P$  is empty and  $X$  is empty then
  | add  $R$  to  $C$  ;                               // A maximal clique has been found
end
else
  | for vertex  $v \in P$  do
  |   BK( $R \cup v, P \cap N_G(v), X \cap N_G(v)$ ) ;
  |    $P = P \cup v$  ;
  |    $X = X \cup v$  ;
  | end
end

```

Algorithm 2: The Bron-Kerbosch Algorithm. The vertices of the input graph are represented by the set P .

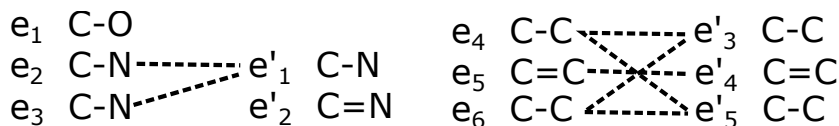
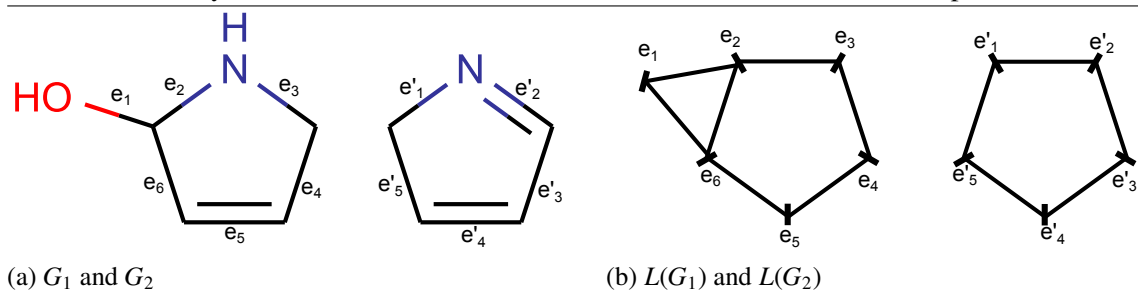
a connected graph of c -edges, thus Koch (2001) applied the modified Bron-Kerbosch algorithm to find these connected c -edges, and then translated the set to the relevant vertices in the parent graph to yield the maximum clique in G (Figure 2.16). The maximum clique of the c -edges, a c -clique, corresponds to the c MCES between the two graphs - a problem which is simpler to solve computationally than finding the d MCES. A c -clique in graph theory terms is actually a path, instead of a clique. Thus, the maximum c -clique is actually the longest path in a modular product (where only c -edges are used). Primarily because of c -edge exploitation, it was found that Koch's algorithms were significantly quicker than the standard Bron-Kerbosch algorithm for finding the MCS between graphs using this simple modification. This algorithm, despite the above description, can be applied to non-isomorphic graphs and thus yields the maximal c -cliques between the two graphs.

2.4.4.2 TOPSIM. The MCES-finding algorithm TOPSIM relies on finding the maximum clique in a reduced compatibility graph of the line graphs of two weighted graphs. The "reduced" compatibility graph is constructed where nodes are made only for matching atom and bond labels. A node in the modular product here is labelled as (e, e') where e is an edge from G_1 and e' is an edge from G_2 of the same edge label, and the vertices on the edge both have the same labels as in e . Edges between these nodes are constructed in the same way as with the modular product, and like the modular product, the maximum clique corresponds to the MCES. Another way of viewing this concept is to think of it as a modular product of two line graphs, but where only nodes are kept which correspond to edges that possess the same properties between the two graphs - edge weights, and vertex labels (Figure 2.15).

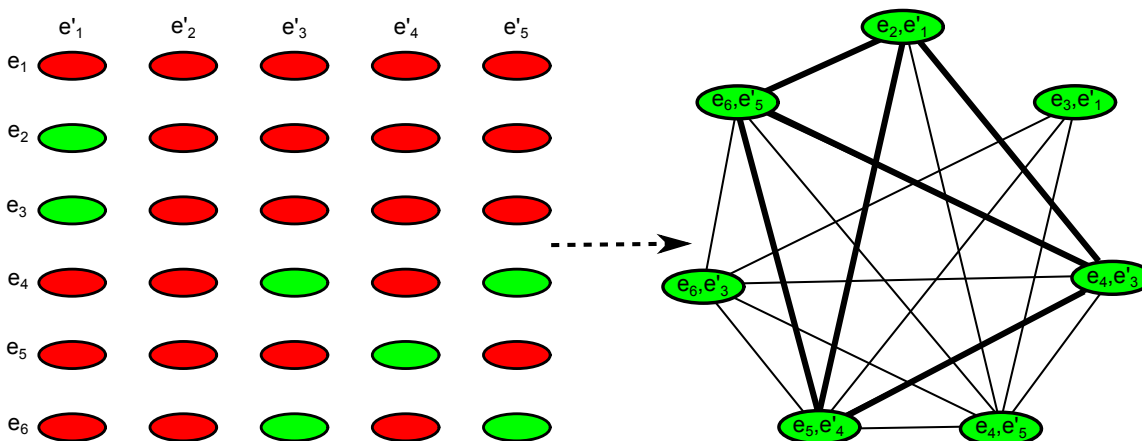
A branch-and-bound algorithm was employed for clique detection in TOPSIM, much like that described in Algorithm 1. Starting from the root, the depth-first algorithm explores each node of the tree (which represents a node in the reduced modular product). The next node chosen in the tree (or rather, the node at the next depth level) is considered legal if it is connected to every other node in the clique identified so far (or if no clique has been found, the first node is selected by default). Once all possible nodes in a path have been explored, the algorithm backtracks, deleting nodes and exploring different paths in the same branch (or perhaps exploring a different branch of the tree). The largest clique found is thus returned upon exhaustion of the entire tree. The lower bound of the algorithm is the maximum clique determined from a pruned version of the reduced modular product (the pruning will not be discussed, though it aimed to preserve edges which had identical neighbouring edges in both graphs). The upper bound by contrast is the maximum possible clique size. (Durand et al., 1999)

2.4.4.3 Babel's Algorithm. Babel (1994) defined a colouring algorithm for finding the maximum weighted clique of a graph. The algorithm starts initially with a random vertex, and then starts assigning colours such that neighbours do not possess the same colour. The next vertex chosen in the graph is the one with the greatest number of colours already assigned to its neighbours, in effect colouring vertices in a localised order of degree. The algorithm used to find the maximal cliques is a *branch and bound* algorithm. Branch and bound algorithms rely on building a search tree where each node represents a potential solution to a problem, and the deeper down the search tree, the more specific the solution becomes. The "bound" part of the name refers to the construction of upper and lower bounds - evaluated solutions are removed if they fall outside these boundaries. "Branch" however refers to the splitting of a solution into a more specific solution set. The upper bound for the Babel algorithm is the largest assigned colour for a vertex, and the lower bound is the weight of the heaviest clique found over the course of the algorithm. The algorithm works on the fact that if a vertex is in a clique then the number of colours assigned to its neighbours must be equal to, or greater than the number of colours in the greatest clique found (otherwise the vertex is eliminated from the graph being studied).

2.4.4.4 RASCAL. Raymond, Gardiner, and Willett (2002b) developed the algorithm RApid Similarity CALculaiton (RASCAL) for both graph-based similarity searching, and MCES detection in chemical structures. At its release date, the algorithm was shown to perform several times quicker than three other algorithms (on a clique-detection level). RASCAL relies on an initial screening process to significantly boost the speed of searching, before a more computationally intensive graph-matching procedure is applied to find the MCES. The initial screening consists of two cost-based distances, which are used as fast-performing screening tiers, so that an MCS search is only performed



(c) Compatible edges of G_1 and G_2 . The set on the left represents the first 3 edge comparisons, whereas the set on the right is the other 3 comparisons. dashed lines indicate an edge correspondence between G_1 (left of each set) to G_2 (right)



(d) The modular product of the line graphs. Nodes in green are compatible (as in same edge and edge-vertex labels between the two graphs, shown in Figure 2.15c). The maximal clique in the reduced modular product (with the shown green nodes) is emphasised with bold lines.

Figure 2.15. the MCES-finding methodology of TOPSIM

when a pair of molecules passes both tiers (in other words, they are similar enough to qualify for an MCS calculation). The first tier involves the calculation of a distance measure between two graphs, based on summation operations of sorted degree sequences of two graphs. The second tier is similar to the first, but instead of degree sequence, vertices are described using edge codes (of each edge incident to the vertex). Edge codes here represent the edge label (which accounts for the fact that chemical graphs are weighted), along with the two vertices on the edge.

The MCS algorithm of RASCAL is a branch-and-bound algorithm for finding cliques in the modular product of two line graphs, with the condition that upon finding the MCES via the clique, the degree

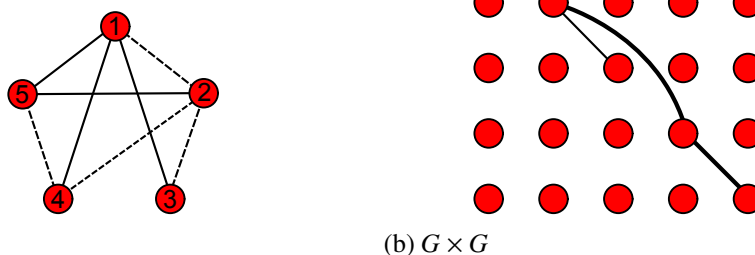


Figure 2.16. 2.16a is a graph G with its c -edges represented as dashed lines. 2.16b is the modular product of G with itself, with the c -edges shown in bold. Note how the c -edges form a path, which when translated to the relevant vertices yields the maximum clique in G .

sequences of the nodes of the relevant subgraphs are compared to investigate if a Δ -Y exchange has occurred. Non-identical degree sequences indicate the presence of such an exchange, thus the clique is ignored for MCES translation.

An alternative form of graph colouring is used, referred to as node partitioning in RASCAL. Nodes in the modular product are initially grouped into colours or “partitions” based on whether they can actually be neighbours with each other. In the case of the modular product, given all the vertex pairs (v_i, v_j) in the modular product, those with the same v_i are grouped in the same partition. Effectively, as graph G_1 would have n atoms, then there would be n partitions to start with. During each iteration of the clique search, vertices remaining in the search tree can be re-assigned into other partitions based on whether they are not neighbours with the existing nodes in a partition. This acts as a fast method of shrinking the “chromatic number” (which also serves as an upper bound), as opposed to re-colouring the graph with each iteration.

RASCAL utilises three different methods for calculating the upper bound bound, based on the size of the candidate set of nodes to add to the clique. During a stage in the search process, the calculated upper bound is calculated as the minimum of these three values:

- If the candidate set possesses less than 150 vertices, then a greedy graph-colouring algorithm is used. The estimated chromatic number $\chi(G)$ is a potential upper bound. Graph colouring has been shown to give a poor estimate of $\chi(G)$ as the size of the graph increased. It was found that a range of 125 to 175 gave a reliable upper bound for this, thus the median of 150 is used (Raymond et al., 2002b).
- If however, the candidate set possesses more than 150 vertices, then a projection bounds method

is used instead, which calculates the upper bound as the sum of all the bond types which are shared by the two graphs, between a given pair of two atoms (e.g. the minimum of each of C-C, C=C, C=O bonds in G_1 or G_2).

- The number of existing partitions (after re-partitioning).

RASCAL also employs two pruning heuristics. Kikusts pruning is performed at each backtrack instance of the algorithm in attempt to remove neighbouring nodes to a clique, which otherwise could not possibly extend the clique. The other heuristic is used whenever the currently explored clique has only one node. This heuristic seeks to find symmetric equivalence classes for the given modular product node in the clique, between the two graphs. Such symmetric nodes are redundant, and thus are pruned from the search. This heuristic is particularly useful in highly symmetric compounds, and can thus hasten the search speed considerably.

Barker et al. (2006) used RASCAL to find the MCS in reduced graphs as a similarity measure. Notably, the MCIS and MCES were used, and two types of reduced graph were made. The first type were “topologically-connected” graphs, where only nodes connected immediately to each other were joined by edges in the reduced graph (in other words, a standard reduced graph). The other was “fully connected,” where all nodes in the reduced graph were connected, and each edge was assigned a weight based on the shortest path length from one node to the other in the topologically-connected graph (essentially, a tdMCS where $\theta = 0$). They noted that the MCIS on the fully-connected reduced graphs yielded the best results in virtual screening. Although this seems surprising given that the MCES is normally assumed to be more suitable as an MCS, the small sizes of the reduced graphs made it less likely that edges (i.e pairs of nodes) would match between the two. Notably, the method was also competitive with Daylight fingerprints in active compound and scaffold recall. A similar piece of work (Takahashi et al., 1992) allowed for overlapping features in the reduced graph, and edges in the fully-connected reduced graphs could have multiple weights (representing multiple paths from one feature to another, instead of simply the shortest path).

2.4.4.5 Iterated Local Search. The maximal clique detection algorithm by Grosso, Locatelli, and Pullan (2008) is an iterated local search procedure, relying heavily on neighbourhood rules to obtain a maximal clique. The algorithm relies on three fundamental aspects: movement operators; node penalisation; and restart rules. A modification of this method has been used by ChemAxon to give a fast-performing dMCES method (Englert, P, Personal Communication, August 2013).

Two operators were mentioned in this clique detection algorithm. The first is an "add" move, which augments the current clique with an immediate neighbouring node not present in the clique. The other operator is a "swap" move, which adds a node not adjacent to exactly 1 other node in the clique, and then removing its neighbours in the clique (there should only be one neighbour to this). This yields an equal-sized clique to the previous, but transformed to a different space in the given graph. Node penalties are assigned to a node whenever it has already been featured in a clique computed from a previous iteration, where the penalty is equivalent to the number of cliques it had featured in. Finally, the restart rule returns a clique to a smaller former state. The simplest restart rule here reverts the clique to a selected node in the entire graph, and its immediate neighbours. A more complex restart rule involves only performing such a restart if the move is guaranteed to remove a minimum fixed number of nodes (set as 4 in this article for optimum performance). This restart rule, combined with the said node penalties and a node selection mechanism that gave priority to nodes with the highest degrees, was claimed to give the best performances. However, the algorithm is non-deterministic, with most stages relying on random selection to some degree, with little mentioned as alternatives to random selection. Despite this caveat, as well as being a maximal clique algorithm (as opposed to maximum), the algorithm obtained a maximum clique on the vast majority of tests in the aforementioned article (Grosso et al., 2008).

2.4.4.6 van Berlo Algorithm. van Berlo, de Groot, Reinders, and de Ridder (2009) extends the Bron-Kerbosch algorithm to find the maximal clique corresponding to the cMCIS and cMCES, in a similar c-edge approach to that used by Koch (2001), as detailed in Figure 2.17. A set of additional heuristics were applied to improve search times. Notably, the upper bound of the algorithm (the maximal clique size) is calculated using bipartite graph-matching of known c-edges and d-edges connected to the nodes in the currently identified clique. All identified c-edges are also added at the same time, rather than one at a time, which the authors claimed to improve search time. In addition, the algorithm orders nodes in the search tree based on their "centrality" (an average of shortest path lengths to every other node in the modular product), where those that are most central are selected first. Another interesting heuristic in their article is the application of 68 "atom-types," which are assigned to atoms in the two molecules. An initial MCS is found by matching based on these atom types (that is, these atom types are assigned as labels). The mapping from this MCS is then used to seed the search for the actual MCS, in which the atomic symbol is used as the label. Interestingly this method was to calculate similarity (using a graph-theoretic equivalent of the Tanimoto coefficient), and was compared with extended connectivity fingerprint similarity. The authors remarked that the

two methods were of comparable performance for predicting gene transcript levels.

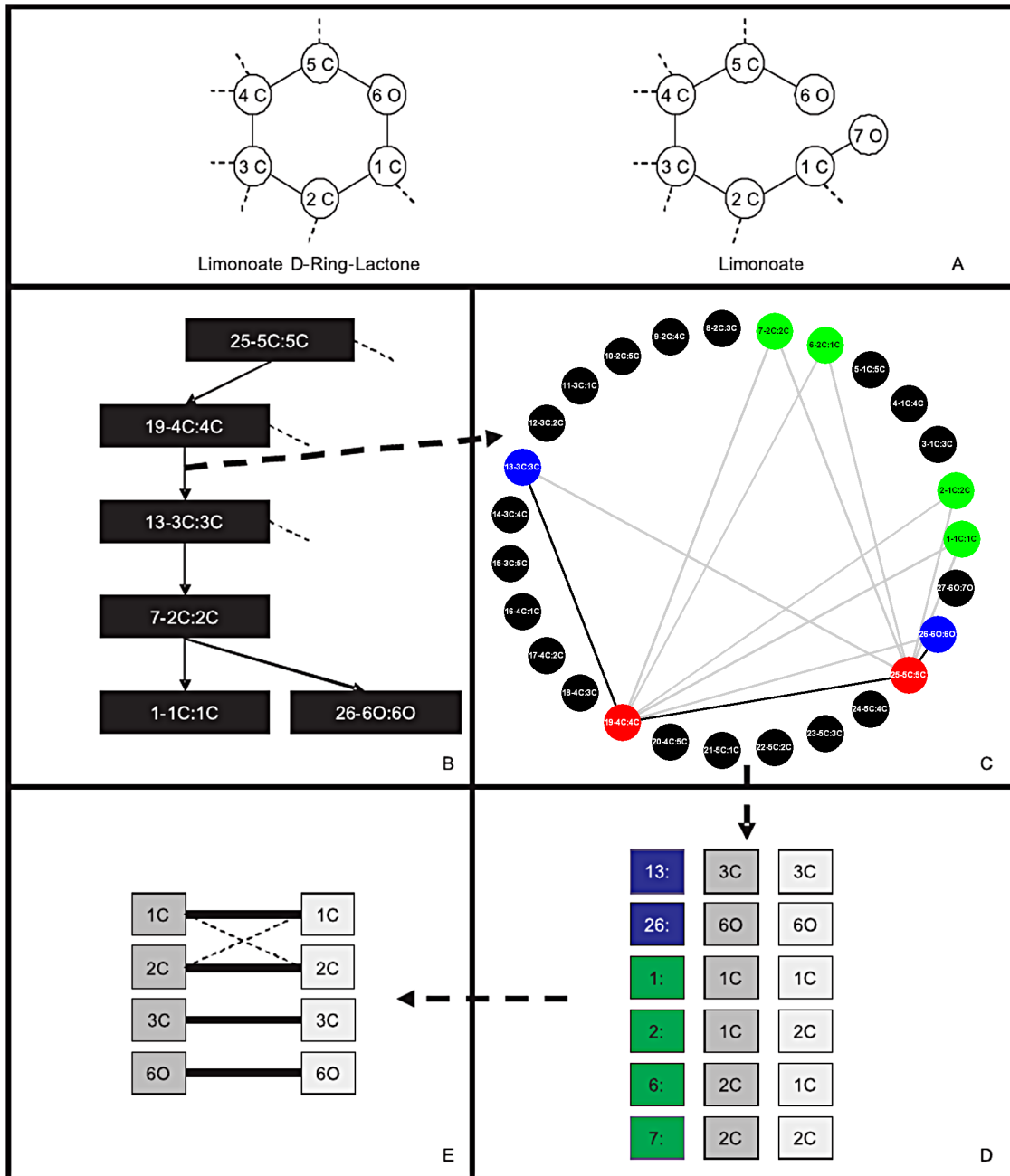


Figure 2.17. Basic methodology of how the van Berlo algorithm finds the cMCIS between the two subgraphs of the molecules depicted in a), using a modular product of the two graphs. b) depicts the search tree for the two graphs. c) depicts the c-nodes (blue nodes, analogous to c-edges from Koch (2001)) and d-nodes (green nodes) in the modular product, when the clique in the search tree has a size of 2. d) shows the c- and d-node matches. A bi-partite matching is constructed from this in e) from which the size of the maximal clique can be estimated (that is, no node from one of the two graphs can feature twice in the maximal clique). The matching here yields 4 edges, which when added to the current clique size of 2 yields an upper bound of 6 for the maximal clique size. Adapted from (van Berlo et al., 2009)

2.4.4.7 MaxCliqueSeq. Depolli, Konc, Rozman, Trobec, and Janežič (2013) for their exact clique detection algorithm “MaxCliqueSeq,” developed a more robust (approximate) colouring method than that of (Babel, 1994) (Figure 2.18). Starting from the first colour (red, in this case), this colour is assigned to the highest-degree vertex (instead of a randomly chosen one, as is the case with the Babel algorithm). Neighbours of this coloured vertex are then removed from a set dictating the vertices to colour, V_{copy} . Once V_{copy} is empty, the set is refilled with uncoloured vertices, and the colour is incremented to the next colour. The next vertex chosen for colouring is the first available in V_{copy} , and the process is then repeated until all vertices are coloured.

In addition to the use of this approximate colouring algorithm, MaxCliqueSeq relies on ordering the vertices in descending order of degree prior to searching for cliques. Interestingly, adjacency matrices and working vertex sets in the algorithm are encoded as bitstrings to hasten their manipulation. The actual algorithm is a depth-first backtracking algorithm, which relies on successive removal of non-neighbours and identically-coloured neighbours to the current clique set (Figure 2.19). A pruning condition is utilised:

$$|U| + |C| \leq |C_{max}|$$

where $|U|$ is the size of the working set of vertices to explore, and $|C|$ is the size of the current clique, and $|C_{max}|$ the size of the largest found clique. Whenever this condition is breached, the algorithm stops exploring that particular branch of the search tree.

2.4.4.8 Englert and Kovacs Heuristics. The work of Englert and Kovács (2015) has been claimed by its authors to be the quickest inexact method for finding the dMCES between two molecules. Most of the information to be presented here has been obtained through e-mail communications with the main author (Englert, P, Personal Communication, August 2013), which complements their recent publication on the workings of their algorithm in 2015 (Englert & Kovács, 2015). The algorithm seeks the maximum clique in the modular product of two graphs using the fast (though inexact) clique detection method of Grosso et al. (2008). A number of heuristics are applied both during and after the clique search, to improve the time performance, and to optimise the MCS returned upon completion of the clique search. Two main heuristics are applied to bias the search space. The first involves the calculation of a “connectivity score” which represents the number of *c-edges* in a candidate node during clique expansion. Nodes with a higher connectivity score are selected, biasing the search to find a less fragmented MCS. The second is a similarity

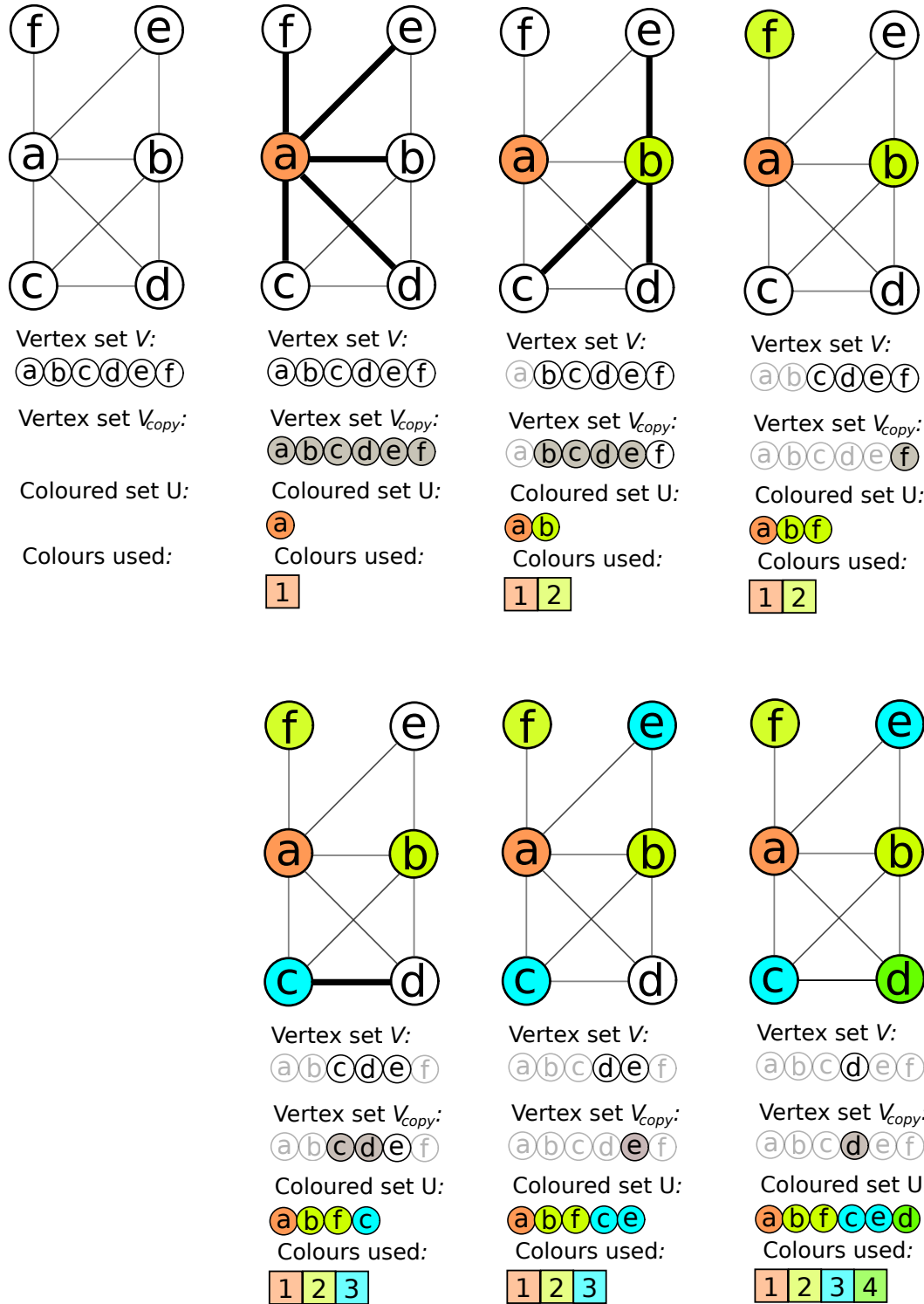


Figure 2.18. Graph colouring algorithm of MaxCliqueSeq, the figure adapted from the same source (Depolli et al., 2013). For V , translucent nodes represent previously-coloured nodes in the graph. For V_{copy} , vertices shaded in grey represent uncoloured neighbours of the vertex being coloured, whereas unshaded vertices represent uncoloured non-neighbours. Translucent vertices here are otherwise un-eligible for consideration for the given stage.

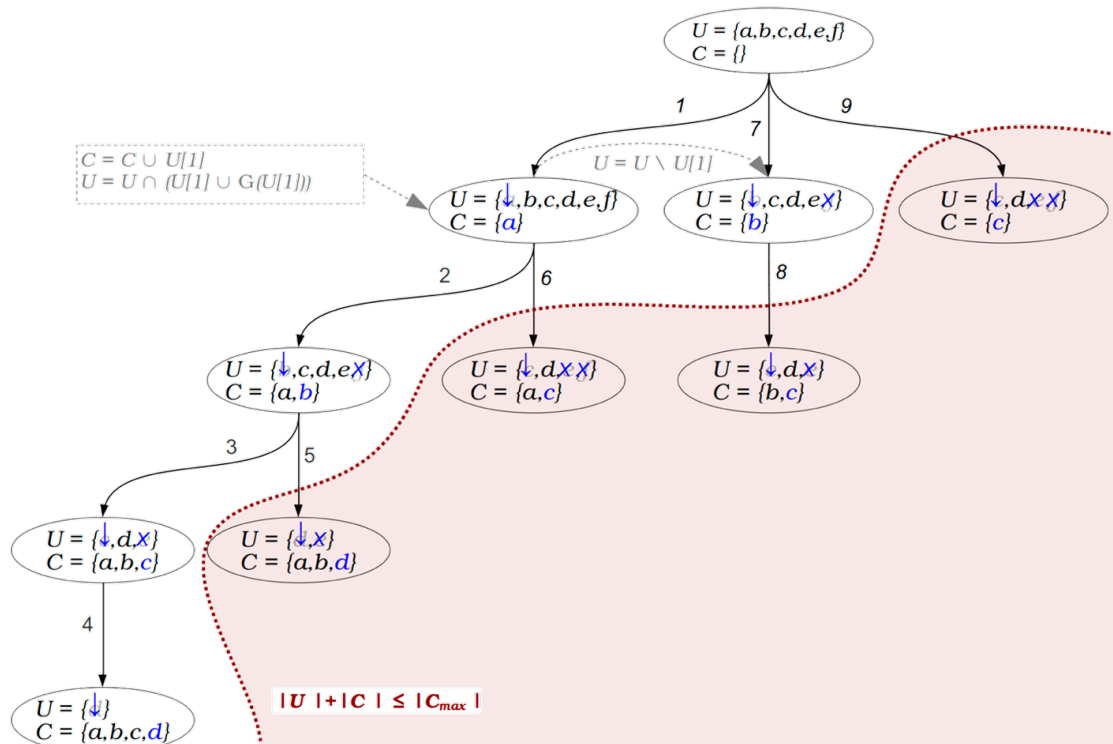


Figure 2.19. Clique detection algorithm of MaxCliqueSeq, the figure adapted from the same source (Depolli et al., 2013), where the graph being searched is that in Figure 2.18. Downward-facing arrows represent the inclusion of a vertex into the clique set C , and crosses represent a vertex's elimination

measure based on the Morgan algorithm, which is used to find estimates for good matching atoms (or bonds) between the two graphs (Morgan, 1965). This similarity measure utilises the Daylight Atomic Invariants rule (Weininger et al., 1989) to define atoms, so in essence is an extension of extended connectivity fingerprinting. The reported similarity is defined as the maximum radius (in topological distance) at which the environments are identical between the two atoms. During clique expansion, nodes are selected in the modular product with the highest weighted sum of the scores from these two heuristics. In addition, the “labelled projection” upper bound for the largest possible size of the dMCES by Raymond, Gardiner, and Willett (2002b) is used as an early termination criterion (the upper bound being defined before clique expansion started). From the author’s tests, this MCES algorithm, with the heuristics, was able to find a maximal clique in just 1000 steps (compared with 100 billion as quoted in Grosso, Locatelli, and Pullan (2008)). Additional heuristics are used to modify the MCS post-clique search to improve the quality of the mapping; the quality score being based on topological distance constraints, and the difference in degree between two bonds.

2.4.5 cMCS via Subgraph Enumeration

One method of finding the cMCS is to enumerate all common subgraphs and identify the biggest one(s) between two molecules. Unsurprisingly, this method can yield the cMCS as the largest fragments in common are identified. An early reported technique in this category is by Armitage and Lynch (1967), a technique proposed originally for reaction mapping as well as for determining chemical similarity (Figure 2.20). In this method, all connected substructures of 2 atoms in length are enumerated between each molecule. Those 2-atom substructures which do not match (atom and bond types) are discarded. The remaining substructures are then extended by one atom, and the atom and bond type filter is applied again. This process is continued until given an extension - no common subgraphs exist, thus all fragments in common are retrieved. This method employs a small number of heuristics, including a canonicalisation step for fragments (which involves checking the atom identifiers of the original fragment against its inverse), and removal of identical fragments (identical in terms of atom identifiers in each molecule, not labels).

Varkony, Shiloach, and Smith (1979) extended the above algorithm for multiple “similarity definitions,” enabling users to produce MCSs which ignored atom and bond types, topology-type matching (i.e. ring atoms could not map to chain atoms) and even to take into account the attachment points of a substructure in a molecule. This algorithm employs additional heuristics to hasten the search. One important heuristic starts the algorithm with the smallest molecule of the two (or more) for enumeration, as the smallest structure generally possesses the least substructures. Canonicalisation in this case uses a key for checking, and duplicate fragments are removed based on isomorphism, instead of their canonical keys. After the fragments are extended, edges that cannot be a part of any larger common subgraph are marked so that they are not extended in future.

The algorithm by Takahashi, Satoh, Suzuki, and Sasaki (1987) largely continued the Varkony algorithm but with a faster method for subgraph isomorphism, employing set reduction. An example is shown in Figure 2.21. The atoms of the structures are represented as bitstrings containing atom, bond type and degree information. This is a particularly quick method for facilitating subgraph isomorphism, given that the subgraphs enumerated were in canonical form. It is extremely quick to use bitwise operators to compare atoms, owing to processors being specifically designed to efficiently handle bitwise logic.

2.4.5.1 CDK Clique Detection Algorithms. The Chemistry Development Kit contains a selection of complementary MCS algorithms. CDK currently features two clique detection algorithms,

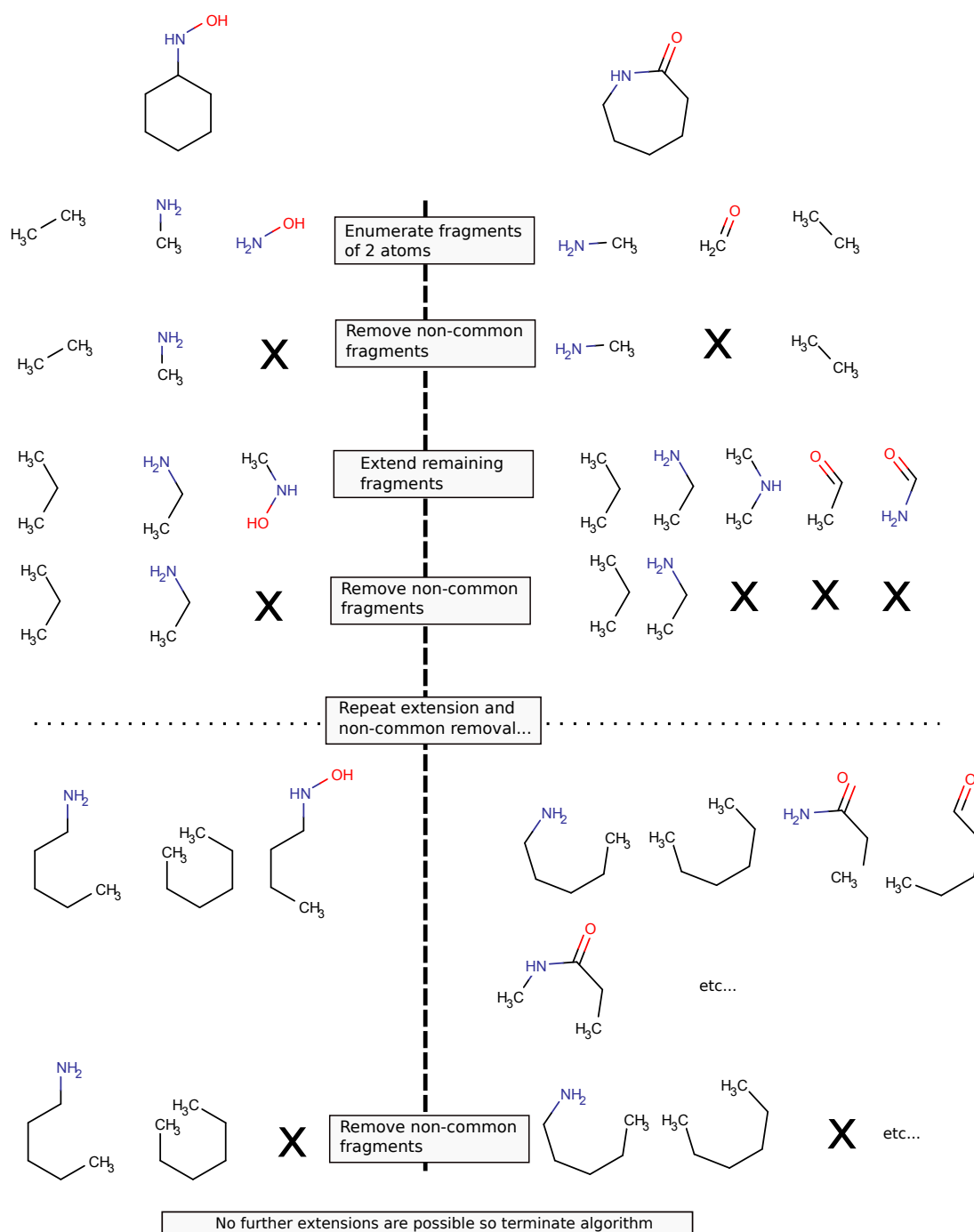
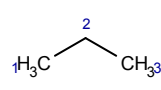
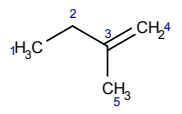


Figure 2.20. A simplified example of the algorithm of Armitage and Lynch (1967) applied to cyclohexone oxime and caprolactam. Fragments do not have atom identifiers shown, and thus identical fragment types are not displayed. This means that although two cMCSs are yielded of different chemical formulae, several mappings involving those two formulae are possible with this algorithm.

CDKMCS and MCS+ (Rahman et al., 2009). The CDKMCS algorithm is an inexact depth-first MCIS search algorithm, which relies on finding the maximum clique in the modular product, in a similar fashion to the Bron-Kerbosch algorithm. Search speed is enhanced using bitstrings to represent the



	atom				bond				degree					
	C	N	O	...	S	D	T	A	1	2	3	4	5	6
v_1	1	0	0	...	1	0	0	0	1	0	0	0	0	0
v_2	1	0	0	...	1	0	0	0	0	1	0	0	0	0
v_3	1	0	0	...	1	0	0	0	1	0	0	0	0	0

(a) Query substructure, v


	atom				bond				degree					
	C	N	O	...	S	D	T	A	1	2	3	4	5	6
u_1	1	0	0	...	1	0	0	0	1	0	0	0	0	0
u_2	1	0	0	...	1	0	0	0	1*	1	0	0	0	0
u_3	1	0	0	...	1	1	0	0	1*	1*	1	0	0	0
u_4	1	0	0	...	0	1	0	0	1	0	0	0	0	0
u_5	1	0	0	...	1	0	0	0	1	0	0	0	0	0

(b) File structure, u

$$v_2 = 100|1000|010000$$

$$u_3 = 100|1100|111000$$

$$v_2 \wedge u_3 = 100|1000|010000 = v_2$$

(c) Relationship

Figure 2.21. Employment of bitstrings to compare atoms. Asterisks in the bitstrings of u show that these bits are set as 1 despite not being otherwise absent features in the molecule, to facilitate the usage of the bitwise and operator with the other molecule - referred to as the “maximum rule.” In 2.21c the application of the bitwise and operator to the two atoms v_2 and u_3 yields a result equivalent to v_2 , thus v_2 is allowed to match u_3 . Adapted from Takahashi et al. (1987)

search tree (being the clique nodes, extension nodes and forbidden nodes per iteration), though it introduces a sharp cut-off of iterations (based on the sum of the atom count in the 2 molecular graphs) to severely reduce computation time. MCS+ as implemented in CDK is the c-clique approach of Koch (2001) with an improved version of the Bron-Kerbosch algorithm (Cazals & Karande, 2008), where the McGregor algorithm is used to extend the solution, to yield the MCS. Often the CDK algorithms have a time threshold, which is related to the sum of the number of atoms in the two molecules being compared. MCS+ relies on said time threshold.

2.4.5.2 VFLib. A unification of VF2 in CDK with other MCS algorithms, called VFLib, was developed in an attempt to improve the size of the MCS retrieved. On execution, VF2 must satisfy one of two conditions:

- find an MCS which is isomorphic to one of the two input molecules
- the time limit (as used in MCS+) is not exceeded

if either of these conditions fail, then the MCS+ and CDKMCS algorithms are both executed separately. The solutions from these are combined, and then extended with the McGregor algorithm in attempt to yield a larger MCS.

A hybrid approach exists in CDK which combines VFLib, MCS+ and CDKMCS algorithms (despite the latter two being used in VFLib anyway), referred to as the DEFAULT method. The two molecular graphs in this method are first tested to see if one is a subgraph of the other. If this is not the case, then CDKMCS is used. If CDKMCS timed out, or if the solution retrieved is not equal to the size of one of the two input molecules, then VFLib is used to calculate the MCS (Rahman et al., 2009).

2.4.5.3 fMCS. A cMCS method that is descended from the Varkony and Takahashi algorithms (Takahashi et al., 1992; Varkony et al., 1979) is fMCS from Dalke Scientific (Dalke, 2013; Dalke & Hastings, 2013). Like the ancestor algorithms, the algorithm first deletes all atoms and bonds that are not in common between the molecules being compared, yielding fragments. The largest fragment from each molecule is taken, and the smallest of these largest fragments is optimistically tested for subgraph isomorphism in the other molecule(s), and then extended. If this fails to yield the cMCS, the method then starts with single bond fragments instead, which are enumerated from the input graphs. Canonicalisation of fragments in this case rely on the Morgan algorithm to hasten the process, and branches of the subgraph extension tree are pruned if they are predicted to have an undesirable predicted maximum extension size.

2.4.6 Other MCS Algorithms

Described in this subsection are algorithms used to find the MCS, which we feel do not fit into other categories previously described.

2.4.6.1 McGregor Algorithm. The McGregor algorithm is another backtracking algorithm initially proposed for reaction mapping purposes, in which atoms and bonds not in common between the two graphs are used to generate the reaction maps (McGregor & Willett, 1981). The algorithm acts to find the MCES in a labelled and weighted graph. It works in a similar way to the Ullmann algorithm, though the match matrix is investigated in the algorithm on the matching of corresponding edges, rather than vertices. The algorithm sets the initial match matrix to have all elements as 1, and progressively sets elements to 0 where one edge in G_1 does not correspond to the equivalent edge in G_2 . Correspondence is checked via the construction of two matrices, one for each graph, with $p \times q$ dimensions - p representing the nodes of the graph and q being the edges. A 1 for position (i, j) in one

of these matrices indicates that node i is connected to edge j . An ordering scheme is also suggested which is used to modify the order of edges explored given the currently found subgraph, such as to maximise the number of corresponding edges between the two graphs as soon as possible. McGregor commented however that maximising the coverage of corresponding edges early on did not always yield the best solution to begin with. Thus, he created the “priority lists” in advance, in an attempt to find which path to explore deep to maximise the chances of finding the MCES. (McGregor, 1982)

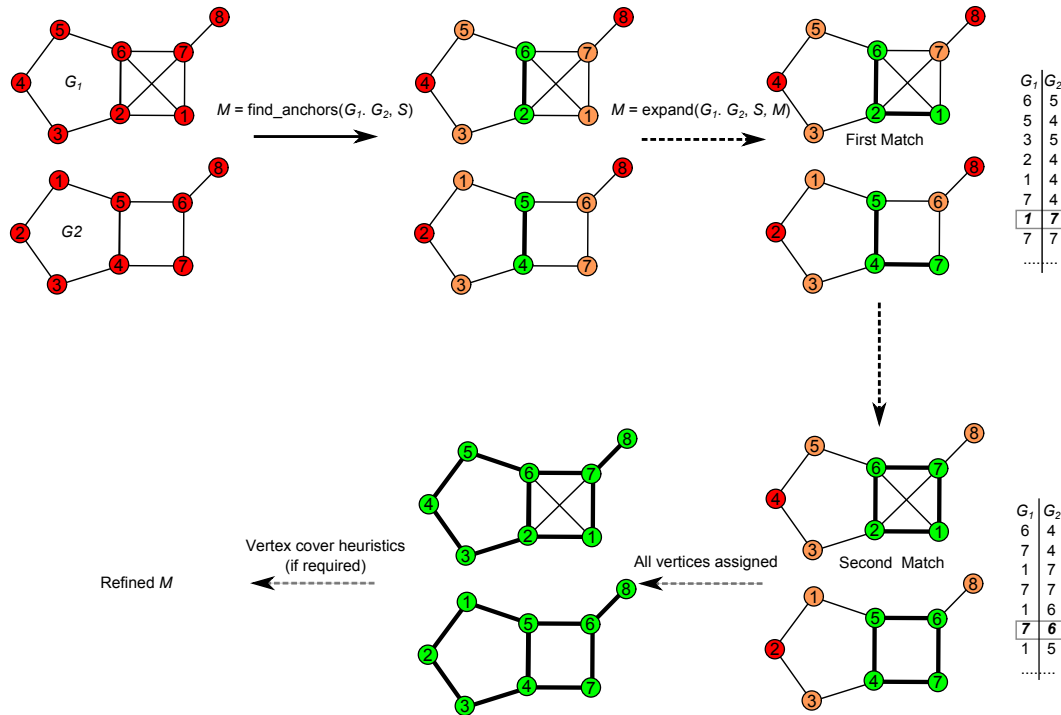


Figure 2.22. An example of the anchor selection and expansion algorithm on two graphs. After calculating the similarity matrix the anchors are assigned, using nodes of high similarity and a greater-than-average degree in the graph (highlighted as the first green vertices, being (u_6, v_5) and (u_2, v_4)). These are used as expansion points for matching the remaining vertices. All possible pairs of neighbours of the two anchors in this case are enumerated and sorted in descending order of similarity. The most similar pair of which possesses vertices which have not already been matched is selected as the new anchor (shown with a grey box surrounding the vertex names for the first and second matches). After all the vertices have been matched (only the first two matching steps being shown), the graph is then passed to the vertex covering heuristics in cases where the match is not optimal to generate the final match matrix.

2.4.6.2 consR. Zhu, Qin, Yu, Ke, and Lin (2011) used an approach termed as *anchor selection and expansion*, which generates a match matrix between two large graphs, in order to find the MCES, with their approximate algorithm “consR.” The anchors here are vertices which possess a high similarity and comparatively large degrees between the two graphs - using a similarity matrix S such that

$$M[u, v] = S_l[u, v] \times S_g[u, v]$$

where S_l is the local similarity matrix and S_g is the global similarity matrix (both similarities explained in more detail in Chapter 3). u and v represent vertices in G_1 and G_2 respectively. Anchors are then expanded to match neighbouring nodes between the two graphs, using the local similarity matrix. The expansion picks the neighbouring vertex sets to the anchors, and generates a list of possible “candidate pairs.” The pair of vertices with the greatest similarity is thus registered in the match matrix, and any other candidate pairs are assigned provided that none of the vertices in the pair have been previously assigned. These assigned vertices then become anchors themselves for which neighbourhood expansion is then repeated, until the whole graph has been analysed (Figure 2.22).

Zhu et al. (2011) implemented a refinement of the match matrix M based on vertex covers. Refinement in this approach relies on attempting to re-match nodes from the two graphs, which are not present in the vertex covers. This is based on the idea that as vertex cover nodes are edge-rich, said nodes are more likely to possess correctly-matched edges than ones outside the covers. Ideally a minimum vertex cover is sought, as a smaller vertex cover yields more vertices to refine. It should be noted however that the finding of a minimum cover is itself an NP-complete problem, so the authors used randomly-determined minimal vertex covers instead for cover-based refinement. Minimal vertex covers cannot have any vertices removed without conflicting with the cover constraints. Due to the random determination of minimal vertex covers, this is one of the only MCS algorithms reviewed here which is non-deterministic.

The authors defined a relationship to relate the two graph covers, which involves constructing a weighted bipartite graph of vertices $V(G_1) - C$ and $V(G_2) - F_2$. F_1 represents the matched vertices in G_1 which are in the cover C . F_2 likewise is the same concept for G_2 . However, we are trying to infer properties from G_2 using only the cover from G_1 . F_2 can therefore be inferred via the relationship $F_2 = M[F_1]$. An edge is drawn between each node of one side of the bipartite graph with each node on the other side, and a weight is allocated to each edge based on the formula

$$w(u, v) = |M[N(u) \cap F_1] \cap (N(v) \cap F_2)|$$

where $N(u)$ represents the immediate neighbouring vertices of vertex u . Weights essentially are assigned based on whether any of the neighbours of u are in the vertex cover, have already been matched (are a member of M), which in turn correspond to any matched neighbours of vertex v in the inferred cover of G_2 . For each node in the smaller of the two sets in the bipartite graph, the greatest weight is chosen to determine which vertex v matches u . The reasoning behind this is better explained in (Zhu

et al., 2011), along with the heuristics used to calculate vertex covers.

2.4.6.3 kCombu. Kawabata (2011) described a “build-up” algorithm referred to as “kCombu” for finding the MCIS, circumventing the potentially costly requirement of constructing and searching a modular product graph used in clique detection MCS algorithms (Figure 2.23). In the context of generating the MCIS, atom pairs are generated (a match of an atom in one graph, with an atom from the other graph) and sorted based on a heuristic score. A fixed number K of correspondences are generated (where the first element is one of the top K pairs), and for each extension of each correspondence, the next best-scoring pair is added. This process continues until all (matchable) atoms have been used. The fact that multiple correspondences are produced, means that up to K MCISs can be obtained (though in practice several of these MCS results can be isomorphic).

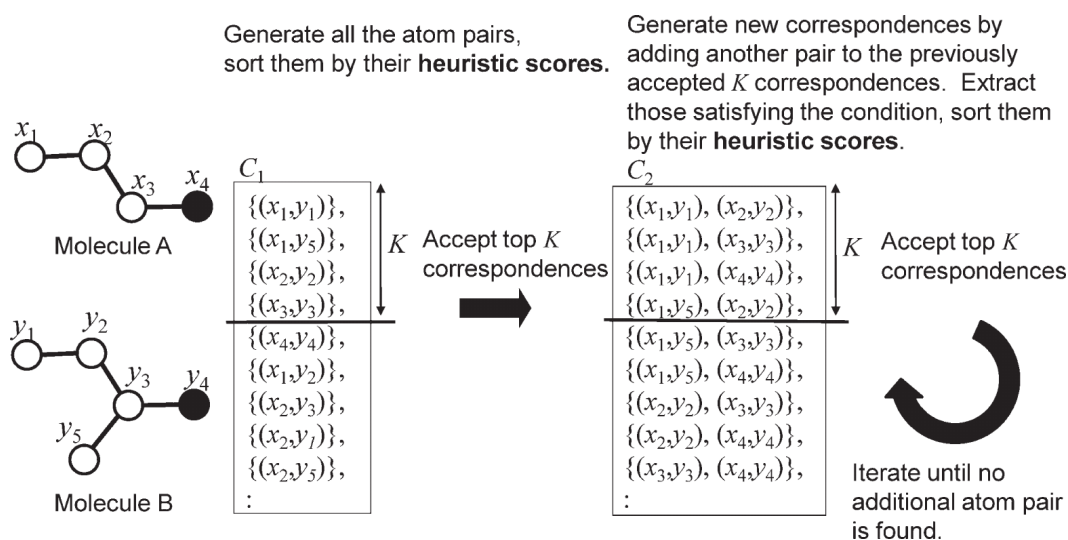


Figure 2.23. Overview of the build-up algorithm, adapted from (Kawabata, 2011).

The heuristic score consists of three distances. Namely, a distance representing the difference in the number of neighbours between two atoms, an extended-connectivity value based on the Morgan algorithm which involves summing up the degrees of neighbours (Figure 2.24), and a topological distance similar to the tDMCS constraint described earlier. The lower the heuristic score, the better a match the atom pairs is.

2.4.6.4 MultiMCS. An exact algorithm proposed by Hariharan et al. (2011) represents a “Divide and Conquer” strategy to finding the dMCS. Briefly, the method relies on enumerating maximal connected subgraphs; in this case using a modified form of the Bron-Kerbosch algorithm to find cliques, although the subgraph enumeration algorithms discussed above can also be adapted for a similar purpose. The maximal subgraphs are then sorted in descending order of size, and then sequentially

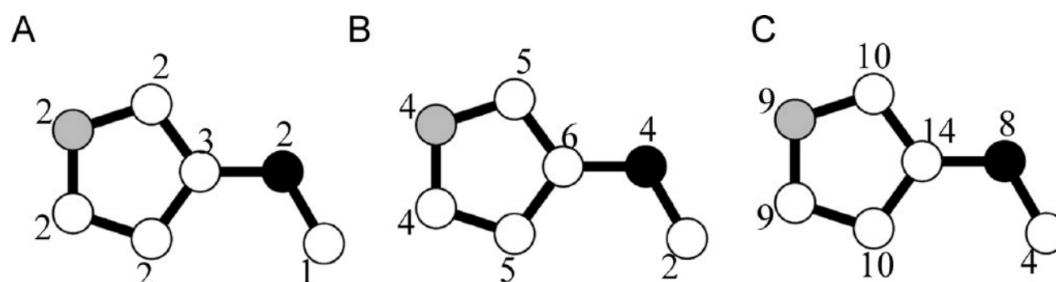


Figure 2.24. The build-up algorithm extended-connectivity heuristic. A represents the first iteration of the algorithm, where each atom is assigned a value equal to the number of heavy atom connections. For the next iteration (B), an atom's value now represents the sum of its neighbours. This is likewise for C, the second iteration. Adapted from (Kawabata, 2011).

intersected. The combination of maximal subgraphs that yields the largest size is deemed the MCS. Although this strategy had a power law in time taken - $O(10^{MCS\ size})$; by breaking down one of the two molecules to be compared into fragments (usually involving the arbitrary breaking of chain-chain bonds), a performance of $O(2^{MCS\ size})$ was obtained.

2.5 Chemical Hyperstructures

2.5.1 Development of Definitions and Properties of Hyperstructures

In translating graph theory to chemistry, an ideal hyperstructure represents the MiCS (Section 2.2.2.2) of a set of chemical graphs. However, the concept of hyperstructures has actually existed before the MiCS was introduced (Bunke et al., 2000). Technically, the first technique in chemoinformatics to resemble hyperstructures is Free-Wilson analysis, an early QSAR technique which relates a bioactivity value to a set of congeneric compounds (a set of similar compounds based on a common lead) based on substitutions of terminal functional groups, around a common scaffold. A simple multivariate regression equation is constructed in Free-Wilson analysis with different weights for the presence or absence of a given substituent (Free & Wilson, 1964).

Perhaps the first work to feature the hyperstructure concept most similar to the work in this thesis, lay in a classification procedure introduced by Menon and Cammarata (1977), having several parallels to Free-Wilson analysis (except that this method was not used for QSAR). A parallel piece of work using the term "hypermolecule" was presented by Simon, Badilescu, and Racovitan (1977), who used a minimisation procedure to create hyperstructures for regression analysis. The term "hyperstructure" was also used to describe a somewhat related piece of work by Mercier, Fabart, Sobel, and Dubois (1991) which first featured in 1973 (Dubois, Laurent, & Aranda, 1973). This latter concept represents

not the MiCS of a set of chemical graphs, but the MCS of all the molecules (called the "focus" in this case), where the rest of the molecule groups were abstracted into nodes forming a tree.

The origins of hyperstructure work at Sheffield lie in a conference proceedings by Vladutz and Gould (1988). The theoretical concept was proposed to increase efficiency of substructure searching, as well as reducing the storage requirements (of the queries). Thus, rather than performing a database search with each individual query structure, the query structures are combined into a hyperstructure by a given method to allow just one database search, typically via subgraph isomorphism of the database structure onto the hyperstructure. This search time improvement was proven to be significantly faster than the total time for the multiple substructure queries, at a later date (R. D. Brown, Downs, Willett, & Cook, 1992).

Vladutz and Gould (1988) gave a somewhat loose definition of a hyperstructure, being a labelled and weighted graph generated by superimposing similar structural graphs overlaying common elements. The node and edge labels represent the source compound(s) from which the node or edge originated. R. D. Brown et al. (1992) presented the more specific definition of "This is a pseudomolecule formed by the superimposition of sets of molecules in such a way that areas of structural commonality are only stored once," which emphasises the point over a reduced storage overhead. The most specific definition to date is from N. Brown, Willett, Wilton, and Lewis (2003), who suggested that a hyperstructure is "a single structure representation of a library, which is generated by the sequential overlapping of each molecular graph in the library to the current hyperstructure. The overlapping is carried out so as to minimise the size (in terms of numbers of nodes and edges) of the resulting hyperstructure." In this thesis, this latest definition will be used.

Due to the nature of the way they are constructed, hyperstructures are not bound by the laws of valence, as shown with the chlorine atoms with 2 connections in Figure 2.25d. Unlike typical chemical graphs, they also often possess a high connectivity (several nodes having high degrees). The latter point in particular has a negative impact on the search speed of graph matching algorithms, due to the greater number of potential subgraphs that could exist between a hyperstructure and another graph.

Hyperstructures may also exhibit the phenomenon of "ghost" substructures - substructures which occur in the hyperstructure, but do not actually appear in any one query individually (Figure 2.25e). Formally, a ghost substructure G_g can be defined such that for the vertices (or edges) $V(G_g)$ of the ghost substructure, $\bigcap_{o \in G_g} = \emptyset$, where o is the set of query structures for the given atom/bond in the hyperstructure that were used to build the given atom/bond. This definition is particularly useful for

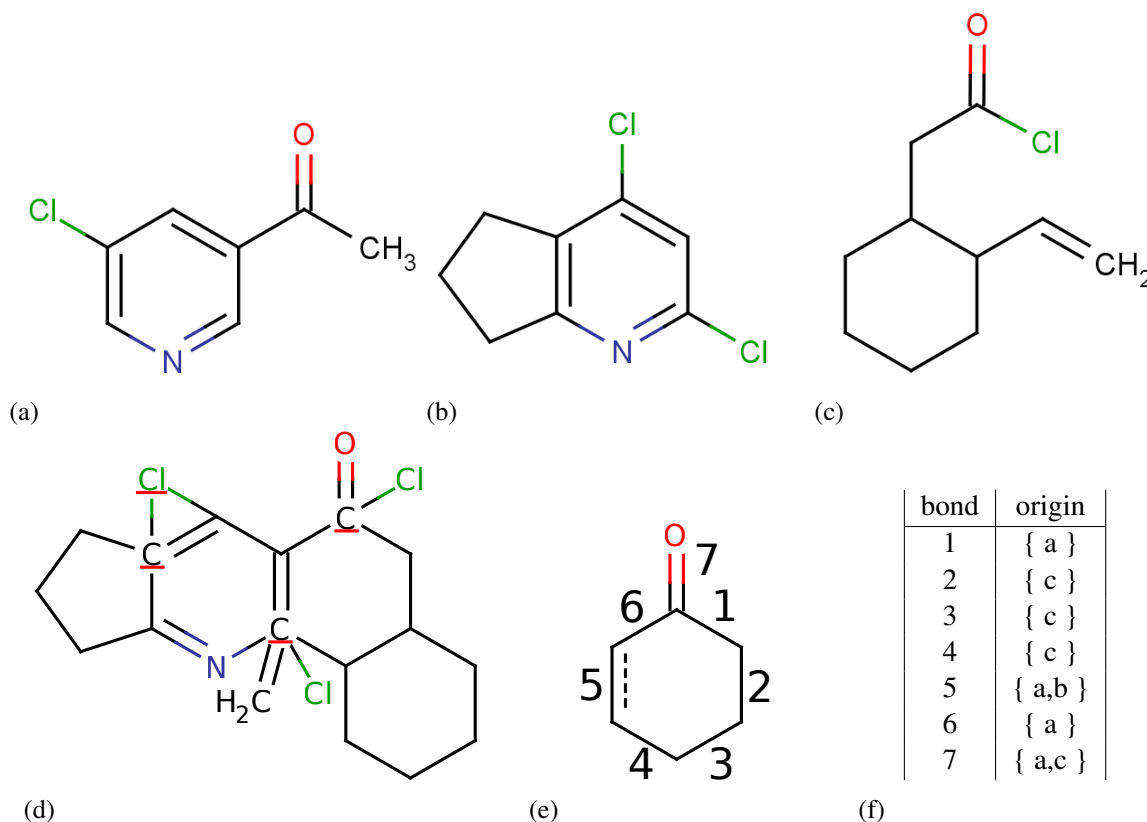


Figure 2.25. Hyperstructure constituents. (a), (b) and (c) represent the starting structures, which are compressed to give the hyperstructure (d). (e) is a possible ghost structure. (f) depicts the query structures that match to each bond of the ghost substructure. Atoms with non-standard valences are underlined in red.

finding ghost substructures in a hyperstructure, for it shows that when applying the Boolean AND to each edge from the matched substructure, a null set is returned if a ghost substructure is found. To give an example using the information in Figure 2.25f starting with bond 7 = {a,c}, if we compare this with bond 6 = {a} then the intersection is {a}. Intersecting this with bond 5 = {a,b} gives {a} as well. However, intersecting this with bond 4 = {c}, then the result is {}, indicating that it is a ghost substructure. It is curious to note that almost no work has gone into investigating the effects of ghost substructures. Vladutz and Gould (1988) employed the Boolean AND method to ensure that only structures found by the hyperstructure matched “genuine” substructures found in the queries. R. D. Brown et al. (1992) likewise avoided the matching of ghost substructures. However, R. D. Brown, Downs, Jones, and Willett (1994) described a screening method for identifying ghost substructures, relying on a dictionary of predefined fragments. Hyperstructures in this work, would only be kept when all the fragments identified existed also in the input molecules. It has been suggested that the effects of ghost substructures are detrimental to virtual screening and substructure searching, but this remains to be proved (N. Brown, 2002).

2.5.2 Construction of Hyperstructures

The NP-complete problem of hyperstructure construction was first approached on a theoretical basis by Vladutz and Gould (1988), who introduced two approximate methods of hyperstructure construction (though it appears that they did not implement them). The first method aims to achieve *maximum overlap* via the stepwise overlap of each molecule onto the growing hyperstructure, essentially in an attempt to find the MiCS. Using the MCES as the initial alignment, the nodes and edges not already in the hyperstructure are appended in some way as to match the topology of the query.

The other approach Vladutz and Gould (1988) suggested, referred to by R. D. Brown et al. (1992) as "stepwise superimposition," starts with generating a giant clique. The clique consists of 100 carbon atoms, and 20 of each heteroatom that the authors assumed would be present in the database structures that the hyperstructure is built from. All the atoms are connected to each other (hence the clique), giving a great number of bonds. Structures are fitted randomly to this via atom-label matching, in order from lowest to highest numbered atoms in the graph (for example, carbons from 1-100, nitrogens 101-120). At the end of hyperstructure generation, those nodes and edges with no origin structures assigned are deleted.

R. D. Brown et al. (1992) implemented and tested variants of these two methods in their work. The maximal overlap method in this work uses the McGregor algorithm (McGregor, 1982) to implement the MCS search. Before graph matching is performed, each node for a molecule has an "environment" matrix calculated, which represents the number of types of atoms a given topological distance from the atom in a molecule. The atoms of a query molecule can then be matched to the structure of interest via environment similarity to speed up finding the MCS. McGregor's algorithm is then used to perform the subgraph matching in a stepwise fashion, increasing a fragment in size by matching nodes adjacent to the fragment produced so far, from the current alignment(s). Being a branch-and-bound algorithm, the upper bound is the maximum number of edge correspondences that could result for each node, and thus the search tree is pruned of paths below this value. It should be noted that, in contrast to the random order of molecules that Vladutz and Gould (1988) used, the authors found that the quickest and most efficient way was to start with the largest structure, to follow with the most similar structure, then to assemble the hyperstructure in this descending list of similarity.

The other implemented method was the atom assignment technique (Figure 2.26), based on the "stepwise superimposition" method. Unlike the original method, however, no bonds are created to start with (or rather, only the atoms are created), so bonds are added during the matching process instead of

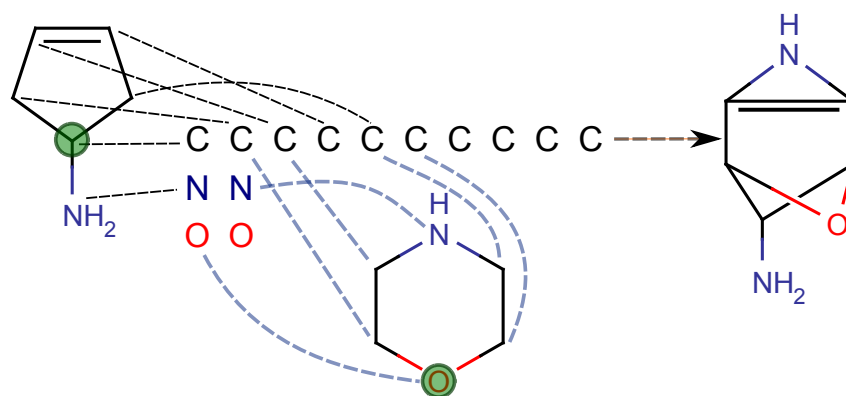


Figure 2.26. The mapping of two molecular graphs onto the pre-defined atoms in the atom assignment method, where a dashed line indicates the mapping (starting from the atom highlighted in green, which in the first graph is the atom with the greatest connectivity). The bonds in the hyperstructure are not shown.

being deleted at the end. Query nodes are also mapped in decreasing order of connectivity, as opposed to a random order. When tested, it was shown that atom assignment only had a minor detriment to compression efficiency compared to the maximal overlap method, despite the significantly smaller construction time.

It will be emphasised that whilst one seeks to find most “compressed” hyperstructure (otherwise the MiCS) of all the input graphs, there is no proof that better compression actually improves the application domain that the hyperstructure is being used for (such as similarity searching). It is assumed here that one should obtain as best a compression as possible to minimise the occurrence of ghost substructures (in particular, large ones). Consider the example in Figure 2.27. Two hyperstructures are created, from two different alignments. Noting that the three constituent structures used to build the hyperstructures are of different ring sizes, the latter hyperstructure has the rings joined side-by-side, as opposed to “in-between” with the former. The former not only has a superior compression, but also a reduced propensity for ghost substructures.

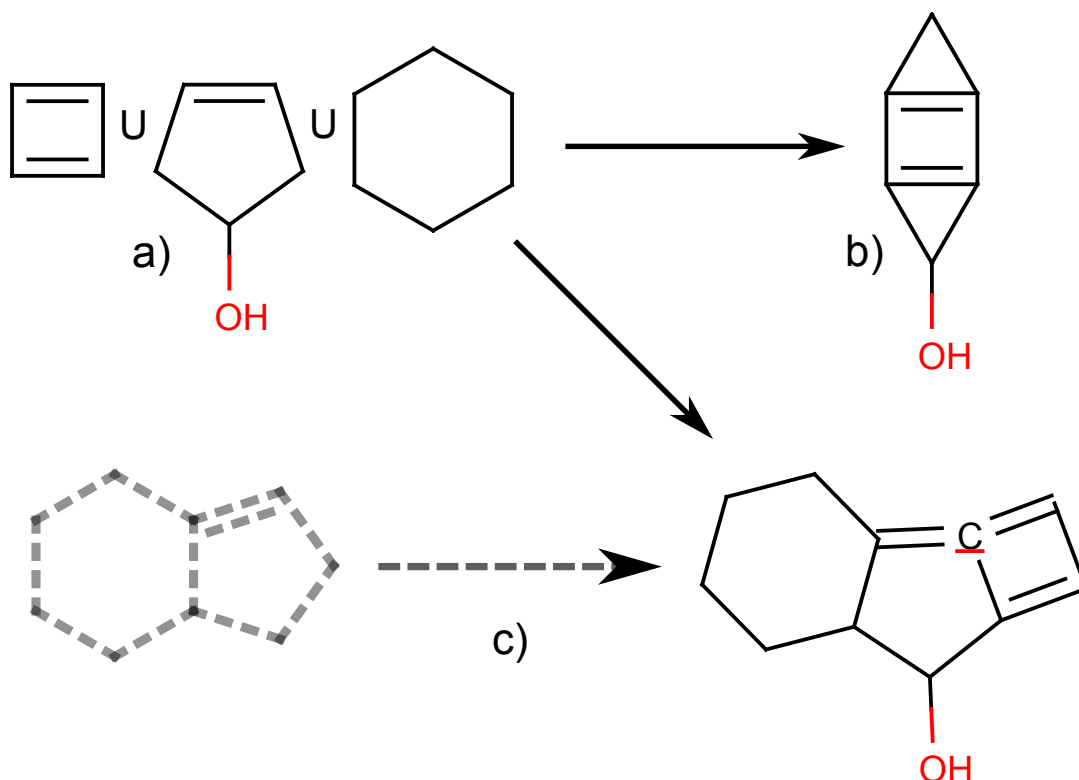


Figure 2.27. The construction of two hyperstructures b) and c) from the query structures in a). b) represents the hyperstructure with the best compression (obtained via MCS), whereas in c), the hyperstructure is constructed using a poorer overlap, resulting in a notably worse node and edge compression than b). The ghost substructure which maps onto c) is shown as a grey dashed molecule.

2.5.2.1 Machine-Learning Algorithms for Hyperstructure Construction. R. D. Brown, Downs, et al. (1994) implemented a Genetic Algorithm (GA) previously employed for graph matching (R. D. Brown, Jones, Willett, & Glen, 1994) in an attempt to create a quicker-performing method than the two exact algorithms just mentioned. Briefly, genetic algorithms are a class of random search algorithms which work on the principle of Darwinian evolution to find an optimal solution to a problem domain. This typically relies on modifying a set of solutions in a series of evolutionary steps, making both small and large adjustments to each solution of each generation, and passing the best solution(s) from each generation.

The said GA uses the dMCES (Section 2.2.2.2) as the template for building hyperstructures from two compounds. Atoms from the query structure are mapped onto the hyperstructure, provided that the atom types are the same, and that the bond between them possess the same label and weight. The fitness between a query and a hyperstructure is the number of correctly matched edges. Once the mapping of optimal fitness had been found, any atoms from the query which don't map are appended to the hyperstructure. When applied to hyperstructure generation, the GA was found to have an almost

5-fold improvement in speed compared to the atom assignment method (R. D. Brown, Downs, et al., 1994).

N. Brown et al. (2003) also applied a GA based on the work of R. D. Brown, Jones, et al. (1994) to generate and search hyperstructures. This algorithm uses Sybyl atom types (Clark, Cramer, & Van Opdenbosch, 1989) instead of simply the elemental types for atom matching. An additional feature added is the application of a meta-evolutionary GA to evolve parameters controlling the population size (the number of chromosomes), and the number of operations that the algorithm runs for. N. Brown (2002) also investigated the effect of compound diversity on the properties of the GA, finding that, perhaps unsurprisingly, compression ratios were worse than when using a random set of compounds of equivalent size.

In contrast to MCS and genetic algorithm-based supergraph construction, Han, Wilson, and Hancock (2015) used an Expectation-Maximisation algorithm to construct supergraphs from a set of input graphs. This work finds the median graph from the training set to initialise the supergraph. The subgraph is then modified for each consecutive input graph by means of minimising a concept called “code-length,” which manifests as a sum of the probability of finding a training set graph in the supergraph, and the von Neumann entropy of the supergraph. This diverges from the previously discussed approach of finding a supergraph of minimal “compression.” Ultimately, this approach was used in both classification of several types of object (including classifying mutagenicity of a compound set), as well as for clustering via PCA.

2.5.3 Predictive Applications of Hyperstructures

The first example of a hyperstructure being used to predict properties was by Menon and Cammarata (1977), using a set of moderately dissimilar drugs from six therapeutic classifications. These were manually and arbitrarily overlapped into a scaffold containing thirteen nodes which represented various functional groups or atoms from each molecule. The 10 most variable of these 13 substituents were converted into standardised molecular refractivities for the given group present from a given input drug molecule (see Figure 2.28 for a simplified example), and the data submitted to Principal Component Analysis (PCA) in a moderately successful attempt to classify the compounds by one of their six therapeutic classifications.

Mercier et al. (1991) used their concept to build regression models to predict glucuronic acid conjugation capacity for a set of alcohols. In this method, each subsequent node along each tree path of

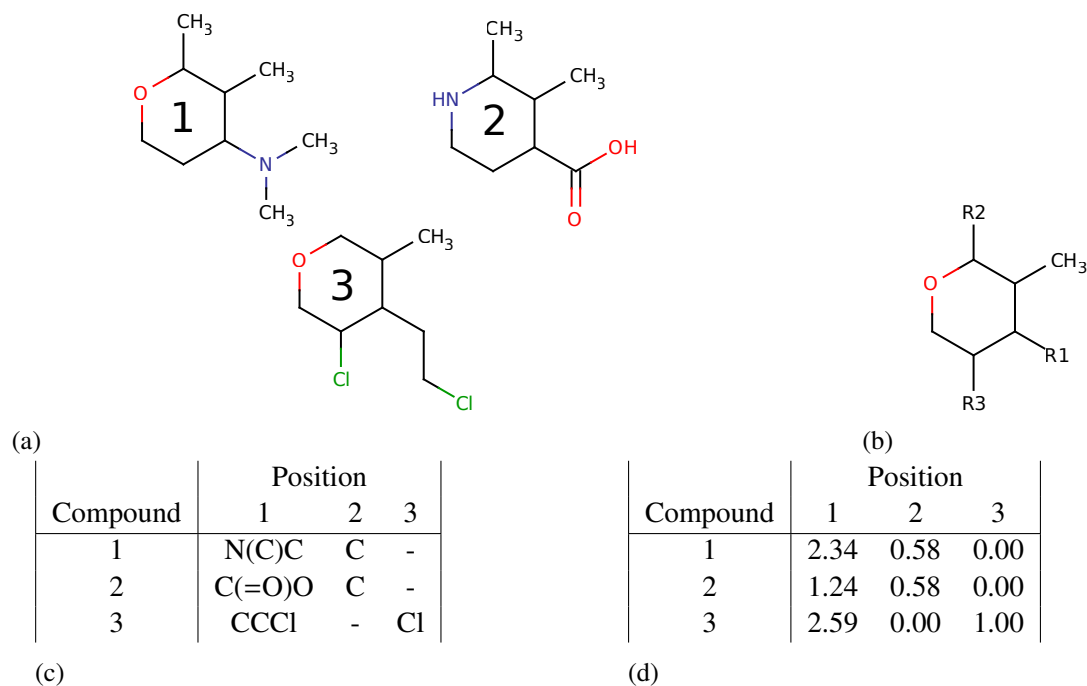


Figure 2.28. A simplified example of the superstructure usage by Menon and Cammarata (1977) resembling the methodology of Free-Wilson analysis. The three compounds in 2.28a are overlapped to yield the superstructure in 2.28b. A tabular representation of the substituents is shown in 2.28c, and the molecular refractivities standardised with respect to chlorine for the groups is shown in 2.28d.

the constructed hyperstructure (from the alcohols) represents a more complex or greater substituent, which in turn is more specific to a lower number of the input molecules. These nodes also contain properties related to the said functional groups which are used to generate regression models. The hyperstructure model marginally outperformed a Free-Wilson model and a Hansch analysis (reduction of compounds to 1D descriptors and building a regression model from these).

Simon et al. (1977) described a process for generating hypermolecules, using a "standard" molecule as the starting point (usually a highly active molecule against the target in question). The construction processes of this method relies on obtaining the best assignments for a molecule's atoms to the given hypermolecule by minimising an error function for activity prediction. Notably, this work uses the Minimal Topological Difference (MTD) approach, which assigns to each node of the hypermolecule, -1 if it is occupied in the standard molecule, +1 if it is not, and 0 if it is irrelevant for steric fit (that is, the node has no atoms from the given molecule assigned to it). The sum of these gives the MTD, which the authors used in a regression equation to model activity.

The hyperstructure concept was further extended to give rise to the Activity-Weighted Chemical Hyperstructure (AWCH). The AWCH is a hyperstructure in which each edge is weighted with an associated activity and inactivity at each edge, these being based on the numbers of active and inactive

molecules which have been mapped to the given edges. It was claimed that AWCHs should have an advantage over techniques such as substructural analysis (Nisius, Vogt, & Bajorath, 2009) due to the ability of the AWCH to better take into account a molecule's topology and the topology's relation to activity. Though, when the AWCH concept was used in a substructural analysis experiment, it was found to be less effective at enriching for active compounds than the 988-bit hashed UNITY fingerprints. The reasons for this were however never explored, for this virtual screening work was performed as an afterthought and did not represent the main focus of the work. One suggested possibility was that "rogue" ghost substructures introduced noise into the searches (N. Brown, 2002).

The work of Klinger and Austin (2006) is essentially a more elaborate investigation into the AWCH, using the Bron-Kerbosch algorithm instead of a genetic algorithm for finding the MCIS (as opposed to the MCES). The MCIS in question is topologically constrained, where the topological distance (θ) is set to 0. The method is concerned with finding the maximum weighted clique when performing similarity searching, and thus uses the Bron-Kerbosch algorithm to find the maximum cliques with the highest sum of weights (the same weight definition as used in the AWCH concept, but for nodes). The authors found that there was no significant difference between weighted and unweighted similarity searching using their hyperstructures. On increasing the number of active compounds in the hyperstructure (up to 5 reference molecules) the recall of the technique improved, though interestingly introducing information from inactives worsened the results. The authors remarked however that the inactive compounds were randomly selected, as opposed to being chosen as near neighbours of the active reference compounds. Other experiments included removing weakly-weighted edges, which had a detrimental effect on the recall (implying that weakly-weighted edges are still important in active compound retrieval), and a comparison with MAX rule MCS group fusion (see chapter 3 for information on group fusion), where no significant difference was found in recall between the two techniques. Considering how extensive this study was - there is an implication that applying simple frequency weighting to hyperstructures is futile, at least as an MCS-based technique.

A related concept to hyperstructures is Molecular Field Topology Analysis (MFTA), introduced by Palyulin, Radchenko, and Zefirov (2000). The concept is related to Comparative Molecular Field Analysis (CoMFA), except that MFTA does not align molecules in 3D. Instead, analogous to the maximum overlap method, it relies on finding the MCS to overlay molecules to generate the hypergraph (in this case using clique detection on the modular product of two graphs). Using a vector to represent descriptors for each atom and bond, a similarity coefficient is used to match atoms and bonds on the two graphs based on these vectors, which was in turn used for both construction and application.

Unlike previous methods described which have relied on screening databases, MFTA is a QSAR method. MFTA uses Partial Least Squares (PLS) regression on derived descriptors from the constructed hypergraph, which are present on the query molecule. Though for the results given MFTA gave more accurate test-set predictions than CoMFA on a set of congeneric compounds, no ability to perform extensive scaffold hopping (or testing it on non-congeneric compounds) was demonstrated. The authors commented however that when they reimplemented the GA by R. D. Brown, Jones, et al. (1994), the GA performed worse than their overlapping method in terms of computation time and compression quality.

A pharmacophore tool which is based on the hyperstructure concept is the Genetic Algorithm with Linear Assignment of Hypermolecular Alignment of Datasets (GALAHAD) (N. Richmond et al., 2006). Briefly, a pharmacophore represents a set of steric and electrostatic constraints which a molecule must satisfy in order to bind to a receptor (Wermuth, Ganellin, Lindberg, & Mitscher, 1998). GALAHAD extends the MFTA method to three dimensions, creating an alternate alignment method called a 3D hypermolecule. Starting from the first molecule, it initially assigns pharmacophore features to the molecule, and flexibly aligns features from this molecule to the features of its nearest neighbour (derived from clustering using pharmacophore features), and continues for the next molecules in descending order of similarity. A Procrustes transformation is then applied to the initial alignment to maximise the overlap (a Procrustes transformation involving movement, rotation and scale transformations, but otherwise no node or edge adjustments). A cost distance-based function is used to calculate alignment quality, and is used to choose those features to remove which contributed to large cost distances, thus generating a reduced feature set. After applying filters to remove geometric incompatibilities, the final pharmacophores can then be applied to test the activity class of test molecules, also allowing the user to select features in the pharmacophore based on how often they appear in the set of molecules used to construct it. GALAHAD was used to align sets of ligands from Protein DataBank (PDB) files (with known binding conformations), and in general managed to reproduce the majority of binding conformations. GALAHAD has recently been used to explain the binding hypothesis of certain small drug libraries (Pizzirani et al., 2008; Zhao, Yuan, Huang, Ji, & Zhu, 2010), and has also been used to guide the development of novel CYP11B2 inhibitors (Lucas et al., 2008).

2.5.4 Caveats with 3D Methods

It is noteworthy to mention that most of the hyperstructure construction methods mentioned so far are in two dimensions. The main reason for keeping to two dimensions is the problem of conformational complexity. Single, non-terminal aliphatic bonds in molecules are rotatable, and thus enable a molecule to change shape to occupy a greater variety of 3D space, as opposed to being rigid and hence possessing one shape. This is sometimes referred to as the *conformational space* of a molecule, and thus must be accounted for when using 3D methods, for the 3D shape of a molecule is key in its interaction with a receptor. A molecule with more rotatable bonds has more possible conformations, and thus is able to access a greater range of conformational space. In receptor-ligand interactions, only a small fraction (if not one) of these conformers represents an optimal shape for a binding interaction. The same applies for molecular alignment - when aligning two molecules it is often the most overlapping conformations of each molecule that give the best alignment. It can be a time consuming process to evaluate each individual conformation in a potentially vast set of conformers. Furthermore, when aligning several molecules the optimal conformation(s) of each molecule must be chosen. Although in most molecular alignment problems it is preferable to select low-energy conformations for alignment (the majority of bound ligands in receptors are of near-minimal energy conformations), several low-energy conformations can still exist for a molecule, which also increases with the number of rotatable bonds. Therefore, the 3D alignment of a hypermolecule is an NP-complete problem that can possess a time requirement far beyond simply aligning the conformations of two molecules. The majority of the methods discussed generally rely on using a large number of molecules to construct hyperstructures, so 3D alignment is somewhat unviable. For GALAHAD (and other pharmacophore construction tools) this problem is simplified as, when applying the pharmacophore, a set of disconnected features is being aligned, as opposed to the atoms and bonds of query molecules. Furthermore only a small number of compounds (i.e. below ten) are used to construct the pharmacophore when alignments are made (A. R. Leach, Gillet, Lewis, & Taylor, 2010).

2.6 Conclusions

This chapter has introduced graph theory in both a mathematical and chemical context, in addition to a number of ways of representing molecular graphs. Exact structure, substructure and similarity searching have been introduced along with methodologies for each, particularly in the context of the MCS. Hyperstructures have also been defined, and the history of their development up to now has

been overviewed.

RASCAL and the Ullmann algorithm are notable benchmarks for comparing with other algorithms for finding the MCS. It is key to note that there is still a need for fast-running algorithms for finding the MCS between graphs, as much work is still underway on both approximate and exact algorithms to date. The work by both Gouda and Hassaan (2012) and Zhu et al. (2011) indicates a movement towards taking into account neighbourhood information of a node in graphs to find the MCS. The majority of MCS algorithms reviewed here are all back-tracking algorithms.

Hyperstructures have been constructed using three methods. The first relies on using the MCES to overlap structures and then append new groups. The second method relies on starting with an excess of atoms of a number of element types, and assigns query structure atoms (and bonds) to the excess and removes any un-used atoms at the end of the procedure. The third method relies on the usage of GAs to construct hyperstructures, based on maximising the number of correctly-matched edges. The latter two methods have been shown to be faster than the first, mainly due to the fact that the then slow MCS-searching procedure was the limiting step. However, both of these methods are also non-deterministic, producing different hyperstructures with each execution. Ghost substructures have so far been neglected in all hyperstructure publications, and it is largely unknown what their effect on database searching is. Most work involving hyperstructures also has primarily relied on using large numbers of molecules to build hyperstructures, and the results when applying the concept to virtual screening are somewhat lacking, with only Palyulin et al. (2000), N. Brown et al. (2003) and Klinger and Austin (2006) attempting to use hyperstructures in some way to enrich for bioactive compounds.

Chapter 3

Similarity Measures in Chemoinformatics

3.1 Introduction

Computational similarity searching has a wide number of applications in chemoinformatics ranging from retrieving analogues of a chemical series, to obtaining novel druglike molecules active against a drug target in *de novo* design. This review will introduce the concept of chemical similarity, with an emphasis on quantifying the similarity between molecules, using 2D graph theory techniques. Chemical space and activity landscapes will be briefly discussed too, with emphasis on methods for visualising similarity. In addition, the concept of data fusion will be introduced and how this fits in with the hyperstructure concept discussed in the chapter 2.

3.2 What is Similarity?

An important point to start with is how similarity is realised in general. Similarity is in fact a highly subjective property depending on what an individual is looking for and the objects being compared. A good example would be to compare an apple, an orange and a basketball, and ask which of these two objects are the most similar? One would perhaps think that as apples and oranges are both fruits they are more similar, yet others would argue that the orange is more similar to the basketball because of their colour and shape. Lin (1998) defines similarity in an Information Retrieval (IR) sense, being that objects are more similar if they have fewer differences between them, and are identical if they possess no differences. Ratcliff and McKoon (1989) in psychology argue that similarity in sentences

relies not only on the shared characteristics but also on word ordering in a sentence. In addition to the sharing of characteristics, Sanfeliu and Fu (1983) proposed another definition of similarity based on edit distance - the number of operations required to change one object into another. An important feature that such similarity definitions have in common is that a similarity measure can consist of three aspects:

- A descriptor, or set of descriptors to represent each object (a number of which have been reviewed in chapter 2)
- A weighting scheme to assign differing degrees of importance to the descriptors used
- A similarity coefficient to provide a quantitative measure based on the proportion of descriptors the objects have in common. (Willett, 2009; Willett & Winterman, 1986)

In chemoinformatics, when molecules are referred to as "similar," it usually refers to a similar topology (or shape, in 3D) and similar content and positioning of functional groups between the two molecules. Put more simply, they look like each other. The importance of similarity in chemoinformatics is expressed in the *similar property principle*, which states that structurally similar molecules have a tendency to exhibit similar properties (including biological activity). A widely cited reference regarding the similar property principle is M. A. Johnson and Maggiora (1990), although similar wordings of this exist in older publications (A. C. Brown & Fraser, 1868; Wilkins & Randić, 1980). This explains why, in general, chemical analogues of a chemical series usually possess comparable properties and bioactivity to other analogues. The Similarity Ensemble Approach (SEA) illustrates a successful application of the similar property principle. SEA in brief performs a similarity search of a test query against a database of drugs with known targets in order to predict the therapeutic properties of the test compound, based on what ligands it is most similar to. This method has been used to predict new drug targets of molecules (Keiser et al., 2009), and also to predict side effects of existing drugs while explaining their mechanism of action (Lounkine et al., 2012). Interestingly, results from SEA show that similarity searching can be used to predict new targets for molecules where their known target has little sequence relationship and no evolutionary history to the new target - the only link being the structural similarity between the ligands of the two targets. Boström, Hogner, and Schmitt (2006) showed, using protein-ligand complexes obtained from the PDB, that similar ligands generally exhibit similar binding modes for a given drug target (based on their relative orientations in the binding sites). It has also been demonstrated that a QSAR model can accurately predict the

activity of a new compound if it is sufficiently similar to the compounds used to build the QSAR model (He & Jurs, 2005).

It should be briefly mentioned that the similar property principle is a rule of thumb and there are always deviations from this. An example is shown in Figure 3.1 with a set of inhibitors against cyclooxygenase 1. Here, a reference inhibitor (Figure 3.1a) is contrasted with an structurally similar but otherwise inactive analogue (Figure 3.1b), yet a dissimilar molecule with a different scaffold possesses potency against the same inhibitor (Figure 3.1c). Cases involving two structurally similar molecules with great differences in bioactivity (as shown in the aforementioned Figure) are termed *activity cliffs*. A typical approach for activity cliff finding is R-group analysis, where chemists deliberately seek a change in one side chain in the scaffold of a molecule to cause a significant change in bioactivity. This, however, will lead to the creation of analogue libraries with a poor chemical diversity and a scarcity of unique scaffolds (Xue & Bajorath, 1999). Readers who are interested in further reading on activity cliff prediction are directed to the review by Guha (2012), with additional examples covered in (Cramer, 2012; Griffen, Leach, Robb, & Warner, 2011).

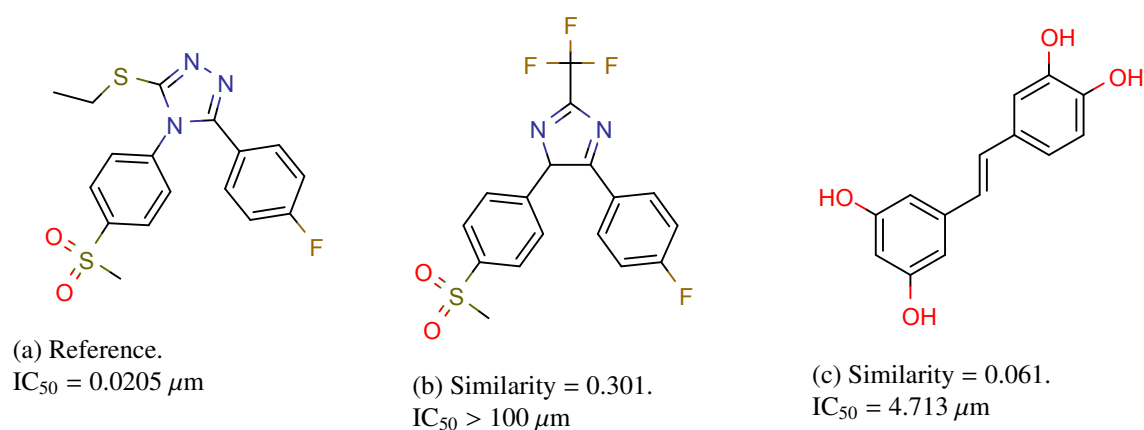


Figure 3.1. A set of Cyclooxygenase 1 inhibitors from (Zarghi & Arfaei, 2011). IC_{50} values are for the COX-1 protein, and similarity values are ECFP6 Tanimoto similarity values to the reference molecule.

Ultimately, this deviation stems from the way that a ligand interacts with a protein binding site, in that often only a few key chemical groups in a molecule actually take part in the binding of a molecule. This can often mean that two molecules with low structural similarity can bind to a target with similar potency, often through the preservation of a small number of functional groups at a fixed geometric distance from each other. Several techniques exist to try to identify such dissimilar molecules, the process of which is often referred to as *scaffold hopping* (Schneider, Schneider, & Renner, 2006; Schuffenhauer, 2012). On the other hand, two very similar molecules may differ only by one atom

(for instance, the presence of a methyl as opposed to an ethyl group) yet possess orders of magnitudes of difference in binding potency. Activity cliffs can have several reasons for their occurrences. The change of a single atom or bond may for instance, completely alter the binding orientation of a compound in its receptor. In less profound cases, replacing polar groups with non-polar groups for example, may significantly alter the activity depending on the original function of the polar group in the receptor.

3.3 Weighting Schemes

The idea of applying weights to modify the results of searches existed well before chemoinformatics. A good example exists in the field of information retrieval, where Spärck Jones (1972) suggested prioritising rare search terms to improve result specificity, and demonstrated that weighted terms significantly improved recall in document searching using key terms. Robertson and Spärck Jones (1976) compared a number of weighting schemes in document retrieval. They noted that two assumptions of independence held - that "the distribution of terms in relevant documents is independent and their distribution in all documents is independent," and that "the distribution of terms in relevant documents is independent and their distribution in non-relevant documents is independent." This was in sharp contrast to the application of different ordering principles, where both the presence and absence of terms from documents made a large difference in accuracy, compared to just the presence of terms.

Willett and Winterman (1986) introduced weighting schemes to chemoinformatics, initially testing three different weighting schemes. The first scheme tested was simply the frequency of a fragment occurring in a molecule. The second scheme was based on the size of a molecule, assigning preference to fragments in smaller molecules. The third was based on the frequency of a fragment in the entire dataset, the scheme assigning preference to uncommon fragments in the dataset. These three weighting schemes were tested using integer counts, and also using binary fingerprints (presence or absence only). It was found that in virtual screening, the difference between binary weighting and counts was profound; frequency weighting generally improving virtual screening yield over binary weighting. However, between the three weighting schemes mentioned (in both binary and counts), there was little difference in results.

More recently, Arif, Holliday, and Willett (2009) experimented with both database and query molecule fragment weighting, notably using the Accelrys ECFC fingerprints (ECFP with frequency weighting), which yielded the best virtual screening results out of all fingerprint types. It was found

that symmetric weighting schemes, where the same weighting schemes are used on both reference and database fingerprints, performed better than asymmetric schemes. Reasons for this related to differences in upper bounds between different weighting schemes. Interestingly, the worse the fingerprint performed in similarity searching, the more the weighting schemes improved their performances, with log-transforming or square-rooting the frequency generally being the best symmetric measure for ECFC fingerprints. This added information on top of what was originally reported by Willett and Winterman (1986) is perhaps why frequency-based weighting schemes are not used commonly when the industry standard of extended-connectivity fingerprints are used, for there is little to no benefit in applying weights to such high-performance fingerprints. Work was also done on inverse weighting, where a rare fragment is ranked preferentially, also in some cases yielding small improvements in virtual screening ability with the ECFC fingerprints (Arif, Holliday, & Willett, 2010). A more recent study by Holliday, Willett, and Xiang (2012) largely confirmed the other studies, though highlighted the Cosine similarity coefficient as being more amenable to weighting schemes than the Tanimoto coefficient.

All weighting in this section thus far has concentrated on frequency weighting, but there is no reason why alternative weighting types can be used, particularly when searching for very specific compounds. One example is by Stiefl and Zaliani (2006), who utilised a node-weighting scheme in reduced graphs based on the 3D structure of the ligand-receptor complex, assigning weights to features where interactions occurred. The weighting scheme was shown to outperform even the FlexX 2.0 docking method (Rarey, Kramer, Lengauer, & Klebe, 1996), in both terms of virtual screening propensity and speed. Furthermore, fusion of the FlexX and weighted reduced graph method improved recall slightly.

3.4 Similarity Coefficients

The similarity coefficient is the measure that is used to compare descriptors and generate a similarity score. Several coefficients exist and have been used for similarity studies long before chemoinformatics, and compared to weighting schemes, an extensive amount of research has been conducted on coefficients. This review will highlight some of the more popular and useful coefficients in chemoinformatics, notably discussing two classes of coefficient - symmetric and asymmetric. Symmetric coefficients give an identical similarity when comparing molecule A with molecule B, to comparing B to A (exchanging the query for the candidate has no effect). Asymmetric, by contrast, may yield

Coefficient	Expression
Tanimoto	$\frac{c}{a + b - c}$
Cosine	$\frac{c}{\sqrt{ab}}$
Russell-Rao	$\frac{c}{n}$
Forbes	$\frac{cn}{ab}$
Fossum	$\frac{n(c - \frac{1}{2})^2}{(c + a)(c + b)}$
Yule	$\frac{nc - ab}{cd + (a - c)(b - c)}$
Simpson	$\frac{c}{\min(a, b)}$
Baroni-Urbani/Buser	$\frac{\sqrt{cd} + c}{\sqrt{cd} + a + b - c}$

Table 3.1

A selection of similarity coefficients which have been used in chemoinformatics. See subsection 3.4.1 for variable definitions

different answers for such an exchange. One should note that this is not an exhaustive review of coefficients. Readers are directed to an article by Todeschini et al. for a comprehensive list of similarity coefficients (Todeschini et al., 2012), and Raymond and Willett for examples of coefficients used in graph similarity (Raymond & Willett, 2002a). Note however, that Todeschini et al. have differing definitions of symmetric and asymmetric coefficients to the terminology used here. A selection of (binary) coefficients mentioned in this thesis is shown in Table 3.1.

3.4.1 Symmetric coefficients

Frequently, molecules are represented as bitstrings in similarity searching. The first molecule for example, has bitstring B_1 with a bits set to “on,” and with B_2 and b being the bitstring and number of “on” bits for the second molecule. B_1 has the same length as B_2 . Arguably the most important coefficient for similarity searching is the Jaccard Index (Jaccard, 1908), which was first developed in comparing biological communities based on their species composition. Mathematically equivalent

(and often far more cited) is the *Tanimoto* coefficient, which is the most popular coefficient for bitstrings (Rogers & Tanimoto, 1960). This coefficient can be represented as

$$S_T = \frac{c}{a + b - c}$$

where

$$c = \sum_i^n \delta_{B_1 B_2 i} n = |B_1| = |B_2| \quad d = n - a - b + c$$

and δ_{ij} represents the Kronecker delta function for two values (bits, for example) i and j

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

More simply put, c is the number of bits that are set as “on” in both molecules, n represents the length of the bitstring, and d represents the number of indices in each bitstring where the given bit is set to “off” in both bitstrings. Bits, for example, could relate to fingerprints, where each bit represents the presence of absence of a particular molecular fragment. This however assumes that the fingerprints of the two molecules are of the same length. It also assumes that a bit of a particular position in the first bitstring corresponds to the same structural feature in the other bitstring, at the same position. When integer arrays (sometimes referred to as counts) are used instead of bitstrings, the parameters for the coefficient are adjusted as

$$a = \sum_i^n x_i^2 \quad b = \sum_i^n y_i^2 \quad c = \sum_i^n x_i y_i$$

where x_i represents the frequency of feature i in the first molecule, and y_i the same for the other molecule. Although this definition assumes that both arrays are of the same size, this can easily be adjusted for when the fingerprints are of different lengths (Horvath, Marcou, & Varnek, 2013).

A related metric in graph matching is the Wallis distance (Wallis, Shoubridge, Kraetz, & Ray, 2001), which uses the MCES as the basis for comparison. The coefficient for graphs G_1 and G_2 is represented as

$$S_W = \frac{|G_{12}|}{|G_1| + |G_2| - |G_{12}|}$$

where $|G_{12}|$ is the number of edges in the MCES. For both coefficients, the closer the similarity is to 1, the more similar the molecules are (1 signifying identity), and *vice versa*. It should be noted

that fingerprint similarity is generally preferable to graph-based similarity in virtual screening, due to its vast improvement in time performance with an almost negligible loss in enrichment over MCES similarity searching (Raymond & Willett, 2002a). The two similarity types however can complement each other, as they have the potential to retrieve different sets of compounds of similar average enrichments (Raymond & Willett, 2002a; Willett, 2009). When using the MCIS instead of the MCES, the values of $|G_1|$, $|G_2|$ and $|G_{12}|$ can be used to represent atom counts instead of bond counts.

Holliday, Salim, Whittle, and Willett (2003) noted that similarity coefficients have an upper bound on similarity which relates to the difference in size between two molecules being compared. They also found that there was a tendency for the Tanimoto coefficient to select larger structures over smaller structures in similarity searching. Although they identified that the Yule and Simpson coefficients are not dependent on molecular size, their relative performance in virtual screening is worse than the Tanimoto coefficient. Interestingly, a modified form of the Russell-Rao coefficient was found to be biased towards the selection of large compounds, as its upper bound of similarity increased as the bit density of the database molecule increased relative to the reference molecule. The opposite was demonstrated for a modified Forbes coefficient, which was found to be biased towards small compounds (Holliday et al., 2003). Haranczyk and Holliday (2008) found that size-biased coefficients were detrimental to clustering compounds, with the Russell-Rao and Forbes (amongst other strongly biased coefficients) causing active compounds to be more spread amongst clusters.

3.4.2 What is the Best Symmetric Coefficient?

The standard symmetric coefficient used is the Tanimoto coefficient, first popularised by Willett and Winterman after demonstrating that it outperformed five other coefficients in similarity searching of a wide variety of small compound sets (Willett & Winterman, 1986). More extensive studies have been performed more recently, using a far greater number of coefficients and compounds in the comparisons, obtaining comparable results. Holliday, Hu., and Willett (2002) for example also demonstrated that the Tanimoto at the time remained the most suitable fingerprint for use in virtual screening.

Selection of diverse compound sets can be as, if not more, important than simply finding a virtual screening method that is accurate at retrieving homogeneous active compounds, even if a severe compromise in activity is required to retrieve a greater diversity of compounds. As inferred in the previous section, choosing a similarity coefficient with a size bias may hinder the retrieval of diverse compounds. Haranczyk and Holliday (2008) found that correlation coefficients generally outperformed association coefficients when used in both clustering and selection of diverse compound sets.

However, the Baroni-Urbani/Buser coefficient exhibited a comparable performance in clustering, and also retrieved a set of compounds with a size distribution most similar to the original dataset when used to select a maximally diverse set of compounds. The simple match coefficient however retrieved the greatest number of Murcko scaffolds. Al Khalifa, Haranczyk, and Holliday (2009) performed a follow-up study involving non-binary coefficients, and found that in contrast to the previous study, size bias in coefficients had no apparent effect in similarity searching and clustering. They identified no clearly superior coefficient in their studies, though they recommended the Tanimoto and Dice coefficients (along with two other association coefficients) as the most versatile in their studies. The Cosine and Fossum coefficients were also notably powerful for use in non-hierarchical clustering.

An extensive study on 44 non-redundant binary coefficients was performed by Todeschini et al. (2012). The Tanimoto coefficient was found to generally perform better than the majority of coefficients. However, two additional similarity coefficients (referred to as CT4 and HL) were found to have slightly (but significantly) better recall rates than the Tanimoto on the MDDR and WOMBAT databases (see Chapter 4 for descriptions of these databases), especially in the case of the latter. Also of significance in this study is a set of multivariate statistics which shows which similarity coefficients are similar to each other in terms of similarity value correlation. Notably, the coefficients could be classified into four groups, which showed clearly in the plots generated from the multivariate statistics. This has potential implications for choosing coefficients for similarity fusion, which will be discussed later in this chapter.

3.4.3 Asymmetric Coefficients

Asymmetric coefficients have been studied less extensively in the literature than symmetric coefficients. One of the most classically cited asymmetric coefficients is the Tversky coefficient, originally proposed in the field of psychology to account for differences in syntax when comparing one object to another (for instance, the idea that the phrase "Turks fight like tigers" is different to the phrase "tigers fight like Turks") (Tversky, 1977). The coefficient can be written as

$$S_{T_v} = \frac{c}{\alpha(a-c) + \beta(b-c) + c} \quad \alpha = 0 \text{ and } \beta = 1 \rightarrow \frac{c}{b} \quad \alpha = 1 \text{ and } \beta = 0 \rightarrow \frac{c}{a}$$

where α and β are floating point values between 0 and 1. In addition, α may be replaced with $1 - \beta$ to normalise the weights. Setting both values to 1 yields the Tanimoto coefficient, thus it is desirable to have the two parameters different to ensure asymmetry, and throughout this thesis, α will thus be

defined as $1 - \beta$ (Wang, Eckert, & Bajorath, 2007). If a represents the database molecule and b the reference molecule, and $\alpha = 1 - \beta$, a value of β close to 1 will yield results akin to substructure similarity, whereas a value closer to 0 will bias the search towards superstructure similarity. An example is illustrated in Figure 3.2.

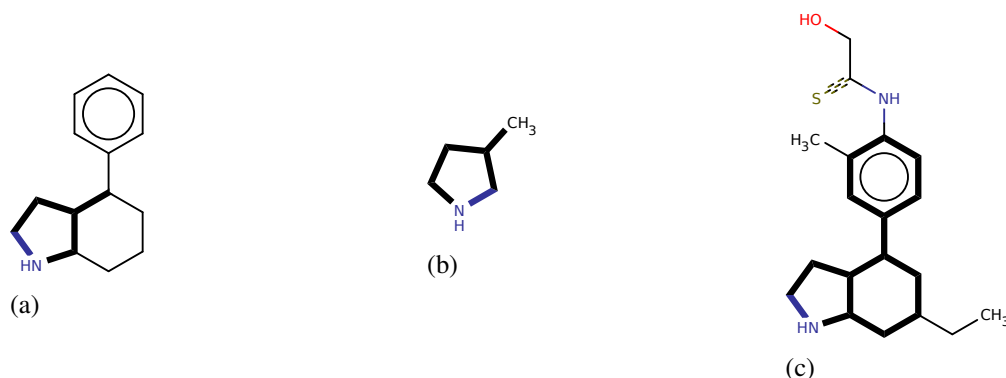


Figure 3.2. An example illustrating the different scenarios arising using the Tversky coefficient, the similarity in this case being the MCES (number of edges in common). Here, molecule (b) is a substructure of molecule (a). (a) has 17 edges, whereas (b) has 6. a represents the number of edges in molecule (a), and b for molecule (b). $c = b$ as molecule (b) is a substructure of (a). Using a β of 0.5, a similarity of 0.522 may be obtained. With a β of 1.0 however, a similarity of 1.0 will be obtained. Another molecule, molecule (c) by contrast is a superstructure of (a) with an extra 8 edges (25 edges in total). If a now represents the number of bonds in molecule (a), and b representing the number of bonds in molecule (c), a β of 0.5 will yield a similarity of 0.810. If β however is 0.0, a superstructure similarity will be obtained, yielding 1.0 as the similarity.

A number of recent studies have been performed on the coefficient to assess the effect of the value of β (quoted as α in these studies) in virtual screening. X. Chen and Brown (2007) showed that putting increasingly high weights on reference compounds yielded improved hit rates over symmetric searches. Wang et al. (2007) demonstrated that bit density has an important role in influencing the results with the Tversky coefficient. When the bit density of the query molecule was higher than the inactive compounds in the selected dataset (assuming actives in the dataset have comparable bit densities to the query), then increasing β decreased similarity values for inactive compounds, for it was more likely that the inactive compound would possess fewer relevant features - and thus were penalised for feature loss. On the other hand, if actives have lower bit densities than inactives, then it is advantageous to lower β to penalise feature gain. A similar issue exists when comparing molecules of different sizes, for a given fixed molecule, it is likely that a larger molecule will exhibit more features in common than a smaller molecule (Wang et al., 2007). Horvath et al. (2013), however, found that a high value of β , 0.9, outperformed symmetric coefficients in every test when screening against various target types in the ChEMBL database, even in the case of feature-sparse sets. They

found that a low value of β was actually worse than a high value in terms of success rate for feature-sparse sets. However, they did not compare the feature densities of the actives with the decoys, though as the same decoys were used in all datasets, it is assumed that the superior performance of the high β value overall can be attributed to actives overall being more feature-rich. Despite this, a low β still marginally outperformed the Tanimoto coefficient on low density datasets, and the two values of β retrieved somewhat complementary sets of compounds. This implies that perhaps both values of β should be used in searching (referring to the Data Fusion section in this chapter). A high value of β is also recommended by Daylight in their internal tests on the Tversky Coefficient (Daylight Chemical Information Systems, 2011) .

Senger (2009) utilised the Tversky coefficient, using Daylight fingerprints as a descriptor, in a substructure-search fashion to perform scaffold hopping on a number of ligands for drug targets. The initial search however retrieved a large number of false positives, so the molecules were fragmented via single bonds and the fragments were used with the Tversky coefficient again to refine the initial list. This fragment-based filtering enabled the author to retrieve several molecules with different scaffolds, on the basis that the other fragments surrounding the core were highly similar.

Unsurprisingly the Tversky coefficient is not the only asymmetric fingerprint that has been studied, though it is easily the most popular. Yoshida, Shida, and Kindo (2001) argued that asymmetric similarity is more suitable than symmetric similarity when comparing documents, proposing a coefficient resembling the Tversky coefficient with an β value of 1.0 or 0.0 (depending on context). B. Cao, Yang, Sun, and Chen (2011) used an asymmetric coefficient to predict the ratings that a user would assign to an item, based on similar ratings by similar users. Mestres and Maggiora (2005) also review a number of additional examples of asymmetric coefficients used in a number of disciplines.

3.5 Graph Theory Similarity

A number of concepts relating to graph similarity were already touched on in the previous chapter. Subgraph isomorphism in particular is of great relevance, in which isomorphic graphs could be regarded as possessing maximum similarity, though several independent methods exist which attempt to quantify graph similarity. Applications of graph similarity outside of chemistry include image analysis (Evans, Wilson, & Hancock, 1995; Han et al., 2015), document clustering (Wang, Ni, Sun, Tong, & Chen, 2011) and biometric identification (Aldridge et al., 2011). This section will attempt to review a small subset of such similarity methods, a more detailed list of which is reviewed by (Conte, Foggia, Sansone, & Vento, 2004).

Typically graph similarity methods rely on two aspects to calculate similarity - a method of matching graphs, and a similarity measure. Tsai and Fu (1979) introduced a cost distance function - a value assigned for the transformation of one node into another. This was based on calculating differences between properties assigned to each node in the two graphs being compared. Ultimately the distance function was used with the maximum likelihood method to generate the best subgraph isomorphism between two graphs. The same authors later extended the concept to allow for node insertion and deletion (Tsai & Fu, 1983).

Sanfeliu and Fu (1983) went on to define two forms of graph distances - feature-based, and cost-based. Feature distance was defined as the application of a measure (such as Euclidean distance) to calculate the distance between vectors of descriptors for the graphs. Cost distance by contrast was defined as the minimum number of operations required to transform one graph into another. It was specified that this could be done either by transforming the first graph directly into the second, or by using an external reference (termed a "grammar" in this work) to mediate the process. In this work, a set of cost functions was devised, along with a grammar, in order to distinguish handwritten characters, using an iterative process in order to find the minimum cost necessary for transformation. Raymond and Willett (2002a) commented that feature-based distances correspond (though are not limited) to the use of fingerprints, whereas cost-based distances correspond to the use of molecular graphs directly.

Although a similar process could be applied with molecular graphs (albeit without the grammar due to the way molecules are readily representable as graphs), it has been shown that the MCS actually is related to cost distance, in that any minimal cost distance computation actually can be used to retrieve the MCS of two graphs, and *vice versa*. MCS is related to edit distance through the relationship

$$d(G_1, G_2) = |E_{G_1}| + |E_{G_2}| - |E_{MCS}|$$

This is a useful relationship, for no cost functions need to be defined, seeing as the MCS is generally not arbitrarily defined. This relationship assumes the application of a certain cost function - where node deletion and insertion have a cost of 1, identical edge and node substitutions have no cost, with otherwise infinite cost if their labels differ (infinite cost meaning they cannot be performed) (Bunke, 1997, 1998). The previous chapter discusses a selection of methods used to determine the MCS of two graphs, the rest of this section focuses on non-MCS methods to determine graph similarity.

Umeyama (1988) introduced a spectral similarity method between two graphs built on the isomor-

phism eigendecomposition method (described in the appendix of A. R. Leach and Gillet (2007a)). By multiplying the eigenvector matrix of one graph with the transposed eigenvector matrix of the another graph with the same number of nodes, a similarity matrix could be obtained: $S = U_1 U_2^T$. Applying the Hungarian algorithm to this matrix (finding the optimum value for each row or column, in this case the highest value), the optimum mapping of one graph to another could be found. Zhu, Qin, Yu, Ke, and Lin (2011) extended this method to allow matching of graphs with different node numbers, which they termed the “global similarity” of two graphs. They found, however, that this method often did not find the optimal node assignment in less similar graphs, thus they introduced the concept of “local similarity”. This was a matrix where nodes were matched based on a normalisation of the degrees of their k-nearest neighbours (a relationship of the sum of degrees in the neighbourhoods, to the number of vertices and edges in the neighbourhoods in the two graphs) - thus nodes in the two graphs with similar neighbourhoods would have higher similarities. The same authors used a method based on these two similarities to find the MCES, as detailed in the previous chapter.

A related concept to graph similarity is molecular alignment, in which the method attempts to find the best superimposition of one molecule onto another. Although in principle a 3D technique, several similarity definitions are of noteworthy mention and are relevant to topological graph techniques. N. J. Richmond, Willett, and Clark (2004) utilised an image recognition-inspired technique to align molecules in 3D. A cost function was defined to assign distances to atom pairs between the two molecules, based on the difference in partial charges, and a histogram binned on geometrical distance to represent local similarity. After applying a small set of filters, including one related to the refinement procedure in Ullmann’s algorithm, the resulting similarity matrix would be subjected to a linear assignment algorithm (such as the Hungarian algorithm) to find the best alignment. It was mentioned that partial charge was used as a descriptor to implicitly represent the connectivity of an atom, in addition to heteroatom pharmacophore type (especially hydrogen bond acceptor/donor strengths). The algorithm was later used in the pharmacophore tool GALAHAD, as explained in the previous chapter.

Berglund and Head (2010) presented another atom assignment method, which in contrast to the work of N. J. Richmond et al. (2004) relied on discrete descriptors. 25 atom types were defined, and each atom in a molecule was described by a bitstring defining the presence or absence of atoms with one of the 25 types, at a given topological distance. For comparing two molecules, the Tanimoto coefficient was used to generate an atom similarity matrix based on these bitstrings. The method employed an anchor-expansion method somewhat reminiscent of the work by Zhu et al. (2011) to

select optimal pairs from the similarity matrix, rather than simply using a linear assignment. In this case, only one anchor was used and multiple runs were performed using different anchors rather than selecting increasing numbers of anchors. An “atom substitution” matrix was also used, which allowed to compute partial similarities between otherwise different bits set.

White and Wilson (2010) constructed a distribution of fragments derived from a set of input molecules. This distribution was used to enumerate all possible new molecules by selecting a scaffold fragment, and linking other fragments systematically at the R-groups on the scaffold fragment. This enumerated “projection set” of molecules was used to create a Gaussian Mixture Model (GMM), using weighted adjacency matrices as the input for the model. The GMM was sampled to create “quasi-molecules,” which were subjected to an alignment with the projection set. The quasi-molecules of sufficient similarity were chosen. This method was used in an attempt to recreate a set of cyclooxygenase-2 actives, and epidermal growth factor receptor actives. Ultimately, the application of FP2 OpenBabel fingerprints (O’Boyle et al., 2011) for ranking the projection set proved superior to this method for finding active compounds, though the graph-based method was shown to sample a diverse range of compounds.

Teixeira and Falcao (2013) also reported an alignment method based on neighbourhoods. Atoms were described via their bond neighbourhoods, where each bond was described through a tuple of continuous descriptors. The similarity between two bonds was a weighted product of these descriptors, which was adjusted accordingly when the bonds are further away from the two atoms being compared. The Hungarian algorithm was then applied to select the maximum bond similarities and these were added to yield a similarity between the two atoms. Once the process had been repeated for all atoms, the Hungarian algorithm was then applied to the atom similarity matrix to yield an atom assignment, which could then be used for similarity calculations with the Tanimoto coefficient. The method was compared with the FP2 fingerprint from Open Babel (O’Boyle et al., 2011), a path-based fingerprint which encapsulates some ring information. Although the method outperformed FP2 in terms of activity class description for a set of 31 steroids, FP2 is not an industry-standard fingerprint and its performance is known to be somewhat inferior to extended connectivity fingerprints, at least according to one QSAR study (Myint, Wang, Tong, & Xie, 2012).

3.6 The Activity Landscape Paradigm

As has been mentioned previously, the similar property principle is a rule of thumb. There are of course instances where molecules of seemingly high structural similarity have great differences in ac-

tivity, the compounds reported in Figure 3.1 being a good example. Such large differences in activity amongst congeneric compounds have been referred to as *activity cliffs*. Such terminology stems from the concept of the *activity landscape*, where the *chemical space* of compounds is typically represented in an N-dimensional space (dimensions typically representing descriptors), with an additional dimension describing the activity. As demonstrated in Figure 3.3, compounds close together in chemical space are regarded as being structurally similar, and the converse of course for those distant from each other in chemical space. A good QSAR or similarity search method will accurately describe the chemical space of the problem domain, including (if not especially) activity cliffs. Activity cliffs are of interest in drug design as the slow and expensive task of finding novel scaffolds can be avoided, by applying a small targeted modification to an existing molecule to significantly improve the activity. Unsurprisingly, several methods have been produced in an attempt to describe and exploit this paradigm (Guha, 2012; Maggiora, 2006).

An important point to mention is the overlap of the landscape concept with that of Matched Molecular Pairs (MMP), where a MMP represents a pair of compounds that differ via a single connected substructure, like a ring or functional group (A. G. Leach et al., 2006). Generally, MMPs are structurally similar, and thus ones with large activity differences represent activity cliffs. The majority of such MMP "cliffs" involve transformations that have no more than one or two atoms difference in the sizes of the substructures. MMPs thus represent a useful complement to fingerprint descriptors in activity space, for instances when fingerprints fail to identify molecules as "similar" which are otherwise defined as MMPs (X. Hu, Hu, Vogt, Stumpfe, & Bajorath, 2012). This chapter will not be reviewing MMPs, thus the reader is directed to a recent review by Dossetter, Griffen, and Leach (2013) for more information on MMPs.

Shanmugasundaram and Maggiora (2001) introduced the Structure-Activity Similarity (SAS) plot as a method of finding activity cliffs in a simple manner (Figure 3.4). The SAS plot is divided into four quarters. Compounds with high structural similarity but low activity similarity (that is, they have large differences in activity) form activity cliffs. Compounds with low structural similarity and high activity similarity by contrast represent similarity cliffs - a concept related to but not identical to scaffold hops (scaffold hops being a change in the core of a molecule, rather than a large similarity difference) (Y. Hu, Stumpfe, & Bajorath, 2011; Iyer, Stumpfe, Vogt, Bajorath, & Maggiora, 2013). Compounds with low structural and activity similarity would be classed as "non-descriptive," and those with high structural and activity similarity would give rise to smooth regions in the landscape. Non-descriptive and smooth regions follow the similar property principle, whereas the other two

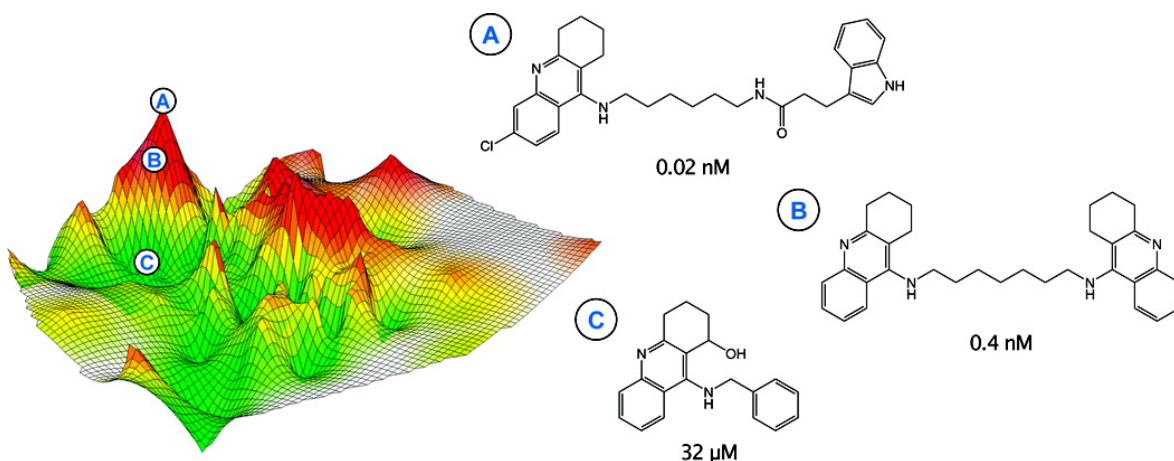


Figure 3.3. Adapted from Peltason, Iyer, and Bajorath (2010). The x and y axes represent chemical space, where molecules close together on the landscape are structurally more similar than distant molecules. The z axis (in addition to colouring) represents activity, where red areas are populated with molecules of high activity and green areas with low activity, and white represents areas which are not populated by molecules. Molecules A, B and C are all structurally similar, yet note the large differences in activity, represented on the landscape by their close positions yet widely differing z-axis values.

classes do not. Activity cliffs have been shown to be the rarest of the four quadrants mentioned (at least when concerning only active compounds), whereas similarity cliffs by contrast are the most common. Both of course are useful, though in quite different ways - for similarity cliffs can often lead to novel scaffolds, yielding otherwise unexplored patent space in drug discovery (Iyer et al., 2013). Guha and Van Drie (2008) introduced the Structure-Activity Landscape Index (SALI) as a quantitative method for finding activity cliffs

$$SALI_{i,j} = \frac{|A_i - A_j|}{1 - sim(i, j)}$$

where i and j are two molecules, A_i is the activity of molecule i (any activity measure, such as an IC_{50}), and $sim(i, j)$ is a similarity function between the two molecules.

Several methods exist for visualising activity landscapes from a set of compounds. Other than the SAS plot mentioned, Guha and Van Drie (2008) described two visualisation methods. The first involved constructing a directed graph where nodes represented compounds, and edges were formed between two compounds if the SALI value is above a threshold. The other method involved creating a SALI matrix of the compounds, and visualising it as a heatmap. Wawer, Peltason, Weskamp, Teckentrup, and Bajorath (2008) created a network graph (termed in their work "network-like similarity graph") where nodes represented compounds, and edges connected nodes if the two compounds formed an

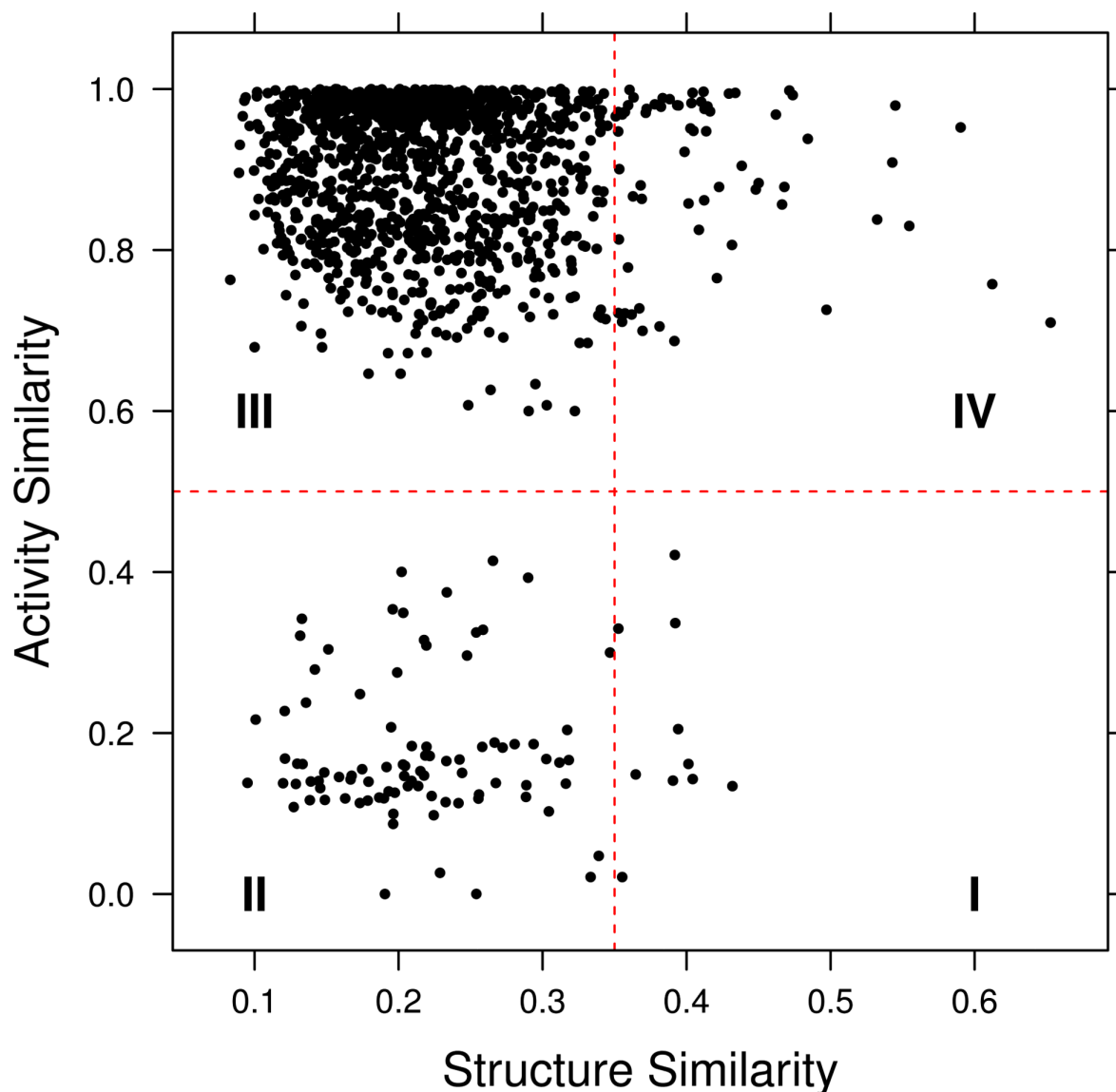


Figure 3.4. Adapted from Guha (2012). The SAS plot here can be divided into four clear distinct quarters, which in numerical order correspond to activity cliffs; non-descriptive; similarity cliffs; and smooth regions.

activity cliff. Interestingly the graph was made viewable in 2D via the application of an energy minimisation technique based on modelling a compromise between electrostatic repulsion (of atoms), and attraction via Hooke's law (acting on edges in the graph) (Fruchterman & Reingold, 1991). Peltason et al. (2010) initially discussed using two descriptors as the x and y axes separately. However, the main body of the work discussed utilising multi-dimensional scaling to generate 2D coordinates for a distance matrix, based on fingerprints calculated for the compounds in the dataset. Activity was applied to the plot as the third axis, a smooth surface being generated through interpolation of the points via the geostatistical technique of Kriging. This technique has been successfully applied in

QSAR and QSPR by Teixeira and Falcao (2014). The models generated were trained using either fingerprints, 1666 e-DRAGON descriptors (Tetko et al., 2005), or their previously published alignment descriptor, overviewed in the previous section (Teixeira & Falcao, 2013). All models were found to be competitive with 3D-QSAR models for predicting Dihydrofolate Reductase Inhibitor activity, noting that using distance-based metrics (as opposed to similarity coefficients) yielded better prediction accuracy. The same technique was applied for aqueous solubility prediction, though was outperformed by previously published (and otherwise unrelated) models.

3.7 Data Fusion

Sheridan and Kearsley (2002) stated that it was not enough to simply rely on one similarity measure alone; there simply is no similarity measure good enough to describe all characteristics of molecules for similarity searching. However, this does not exclude the combination of different sources of information to improve accuracy. This is data fusion in its basic form. The term “data fusion” in computer science originated in the 1980s in the United States’ Department of Defense, who sponsored its development from which it emerged into its own discipline (Hall & Llinas, 1997). Data fusion in a non-computational sense is a very old technique, one good example being trial by jury (twelve independent decisions being used for a conviction, as opposed to just one). Its usage in chemoinformatics is somewhat more recent, a classic example of which is *consensus scoring* in docking applications (and other structure-based methods), which often involves combining several scoring functions to improve accuracy (Feher, 2006). This section however is more concerned with data fusion for ligand-based methods, a few of many examples which will be described here.

Data fusion in chemoinformatics can be divided into two areas - *similarity fusion*, and *group fusion*. Similarity fusion involves the fusion of rankings obtained from different similarity searches using the same reference structure. Group fusion on the other hand uses multiple reference structures using the same similarity measure, where a similarity search is performed for each reference structure and the rankings/similarities are fused to obtain a consensus (Whittle, Gillet, Willett, Alex, & Loesel, 2004). One should note that there is a wide array of potential fusion rules that can be used, ranging from simple statistics (such as taking the mean average of similarity scores) to machine learning techniques or utilising complex multivariate statistics (Willett, 2013). Fusion rules that have typically been used at Sheffield are the SUM and MAX rules, the former being the addition of all the similarity scores (or ranks) for a compound, and the latter being the selection of the highest score (or lowest rank if the compounds are sorted in descending order of score) for a given compound.

The first examples of similarity fusion that appeared in chemoinformatics were at research groups in Merck and Sheffield. Kearsley et al. (1996) and Sheridan, Miller, Underwood, and Kearsley (1996) described systems for their in-house screening system which utilised 2D and 3D fingerprints with chemical probe information, yielding various combinations of similarity searches. Little emphasis was put into interpreting the significance of their data fusion methods, though they noted that fusion was generally (though not consistently) beneficial, despite also noting that the rankings of their geometric (3D) and topological (2D) methods were highly correlated. Many properties and relationships are now known in data fusion, some of which will be described here.

Holliday et al. (2002) demonstrated effective similarity fusion through the SUM fusion of up to three different similarity coefficients, taking each coefficient from different groups after performing cluster analysis to identify complementary coefficients. It has been shown that similarity fusion is more effective when the active molecules used are strongly clustered in chemical space, compared to the inactive molecules (Whittle et al., 2004). Whittle, Gillet, Willett, and Loesel (2006b) and Whittle, Gillet, Willett, and Loesel (2006a) also performed extensive studies into similarity fusion. They noted that similarity fusion using both different similarity coefficients and different fingerprints gave potentially useful fusions, based on an upper bound calculation from the ranked compound lists. In practice, however, the fusion rules used (amongst them MAX and SUM) failed to yield consistent improvements over conventional similarity searching, in contrast to group fusion, which consistently outperformed searching with one reference structure. It was noted that data fusion generally works better when the ranked compound lists for each search are weakly correlated, providing more complementary information sources. A good example of this was shown with the Russell-Rao and Forbes coefficients, two coefficients which when fused demonstrated a notable improvement in recall. The MAX fusion rule on the scores was shown to generally be the best fusion rule for group fusion, whereas the sum of ranks is the best rule for similarity fusion. In group fusion-based virtual screening, a technique called reciprocal rank data fusion was found to be consistently (albeit only slightly) superior to MAX and SUM fusion (tested on both similarity scores and ranks), amongst several other fusion rules tested. Reciprocal rank can be defined for a given database molecule d , as

$$\sum_{i=1}^p \frac{1}{ri(d)}$$

where $ri()$ is a function that returns the rank of the database molecule, for the similarity search against reference molecule ri , where there are p reference molecules (B. Chen, Mueller, & Willett, 2010).

Pareto ranking is a technique introduced into chemoinformatics, primarily used as a fitness function for evolutionary algorithms. It has been popular in compound library design, where it can be used in multiobjective optimisation to select the best compromises between several desirable variables (Ekins, Honeycutt, & Metz, 2010; Gillet, Khatib, Willett, Fleming, & Green, 2002). Pareto ranking in brief, attempts to select non-dominated solutions when optimising for multiple objectives. A non-dominated solution is a data point (a compound in this case), for which moving to another point in the problem space is guaranteed to result in a decrease in at least one objective. At the same time however, this solution has no other data point that is better than it in all objectives (Gillet et al., 2002). Cross, Baroni, Carosati, Benedetti, and Clementi (2010) utilised Pareto ranking as a form of group fusion, between receptor active site, and ligand-weighted GRID molecular interaction fields (MIF). Pareto ranking was shown to be generally the best group fusion, in combination with a form of similarity fusion between the receptor-based and ligand-based MIFs. It was generally best for overall accuracy, as well as in virtual screening and chemotype retrieval. To note, two forms of Pareto ranking were used; the first breaking ties after one iteration of ranking with the SUM rule, and the other method relying on two more iterations of Pareto ranking (using one molecule less with each iteration) to increase ranking precision, remaining ties again being broken with the SUM rule. The latter was shown to be slightly better than the former. Pareto ranking was also shown to be more accurate and robust than the SUM rule in similarity fusion of 3D structure and ligand-based methods, attributed to the fact that it chooses only the best-ranked from each list rather than looking at all the ranks (both high and low) for each compound (Svensson, Karlén, & Sköld, 2012).

Turbo similarity searching (TSS) represents a special form of group fusion. Introduced by Hert et al. (2005), the concept relies on performing a similarity search with one reference molecule, and then selecting its nearest neighbours from the search and performing searches for each (and thus fusing the results as in group fusion). This technique relies on the assumption of the similar property principle, being that the nearest neighbours of the reference compound are also assumed to be active. This assumption proves to hold in general, as Figure 3.5 demonstrates, noting that the top 20 in this case have at least a 50% chance of being active. Indeed, they found that using around 100 nearest neighbours increasingly improved the efficacy of fusion, though adding more than this tended to drop the accuracy with the inclusion of more inactives. The same phenomenon was also observed with scaffold hopping, where increasing the number of nearest neighbours improved framework retrieval. Hert et al. (2006) extended this work by applying the concept to binary kernel discrimination and sub-structural analysis (machine-learning techniques), and showed that these machine learning techniques

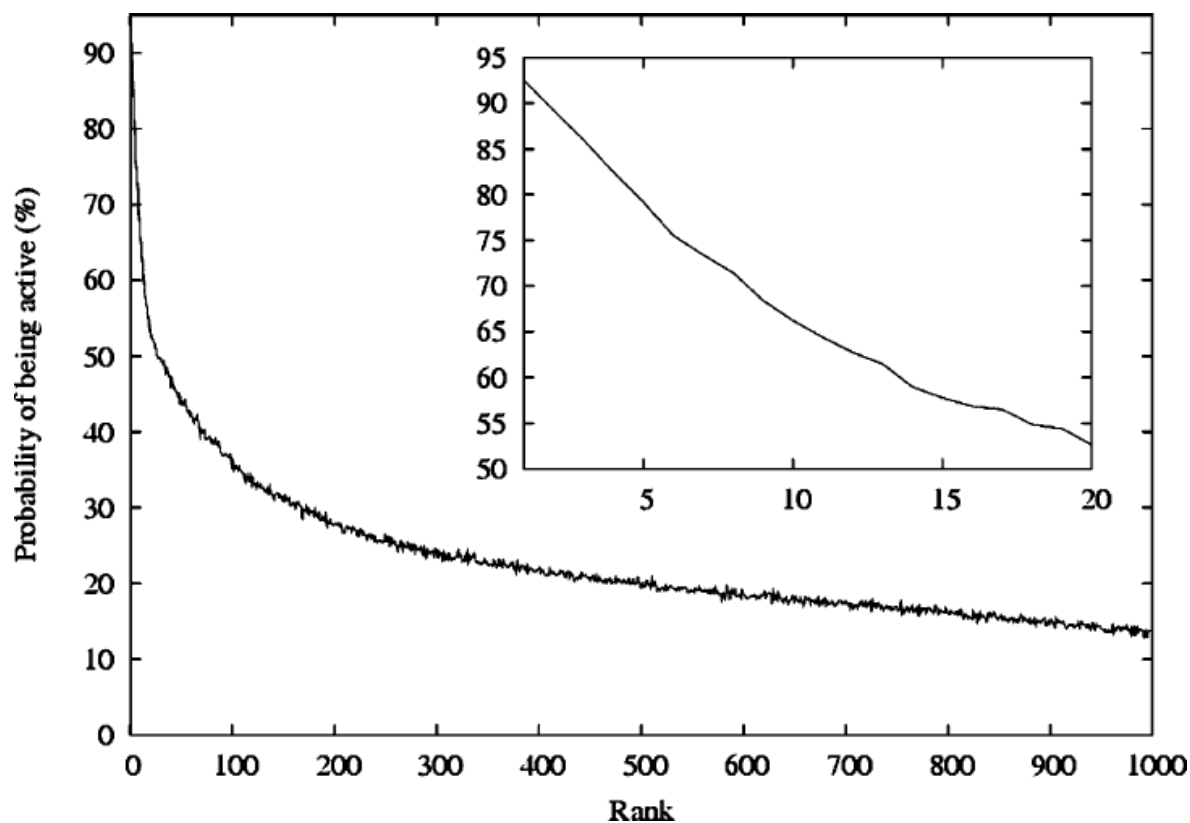


Figure 3.5. Adapted from Figure 2 in (Hert et al., 2005). This graph represents the average probability of a compound being active with respect to its rank. The inset graph is a higher-detail snapshot of the first 20 ranks.

were generally inferior to standard group fusion, though tended to outperform group fusion when the activity class was highly dissimilar. Arif, Hert, Holliday, Malim, and Willett (2009) extended this work by applying the concept to fragment weighting, which largely confirmed the results of the previous findings on turbo similarity searching, finding also that the square root in frequency weighting was the best weighting scheme. Gardiner et al. (2009) performed extensive testing using TSS, using several fingerprint types, numerical and topological descriptors, and reduced graphs. In particular it was noted that TSS was beneficial in homogeneous compound sets, using higher-performance fingerprints, and reduced graphs. The concept was extended to substructural analysis, where it was shown that conversely the technique was beneficial on diverse sets but not so much on homogeneous sets. The same authors also showed that inclusion of a small number of nearest neighbours gave beneficial results when predicting aqueous solubility using extended connectivity fingerprints.

3.7.1 Why Data Fusion is Believed to Work

The experimental rationale for why data fusion works can be traced back to recent work in Information Retrieval. Spoerri (2008) applied several different search systems to seek relevant documents in text-based datasets. It was noted that ranked lists from each search were complementary, though the existing overlap proved to be advantageous - the percentage of relevant documents retrieved increased exponentially with the number of searches to be fused. This yielded a strong power law model (though the relationship is actually sigmoidal as the accuracy must plateau as it reaches 100%). This finding was referred to as the *Authority effect*, and Spoerri (2008) recommended that a good data fusion rule should be used to exploit this. Holliday, Kanoulas, Malim, and Willett (2011) performed related work to show that the concept also worked in chemoinformatics, also demonstrating the power law when applied to similarity fusion. Interestingly the authority effect in group fusion was shown to be more pronounced than in similarity fusion, but was shown to be parabolic (that is, there was an optimal number of molecules to use, instead of a power law), the optimum value supposedly being somewhat dependent on the diversity of the dataset used.

To complement this work, Whittle et al. (2006a) performed an experimental analysis also on similarity and group fusions. It was shown that fusion of uncorrelated similarity coefficients (based on Pearson's correlation of matched pairs in the top 1000 compounds of the ranked lists) tended to improve data fusion potential, especially the Russell-Rao and Forbes coefficients, whose size-bias behaviours are otherwise opposite (Holliday et al., 2003). Group fusion was shown to generally be superior to similarity fusion, in particular when a diverse set of molecules is used as references (again, to have more complementary sources of information). The SUM fusion rule was identified as the best for similarity fusion, whereas the MAX rule was best for group fusion. Related theoretical work mentioned that predicting the precise nature of data fusion is non-trivial, relying on eight different distributions for the fusion of just two ranked lists (Whittle et al., 2006b).

3.8 Conclusion

As mentioned in this chapter, similarity is highly subjective and many methods exist for measuring it, within and without chemoinformatics. From an information retrieval context, similarity typically relies on a weighting scheme, a descriptor, and a coefficient - established methods existing for each of the three categories, though the choice of method varies depending on the desired outcome. Although log-transforming or square-rooting the frequency of fingerprint bits has generally been shown

to consistently improve virtual screening accuracy, the magnitude of improvement depends on the fingerprint (extended connectivity for example being improved only slightly). Similarly, though the Tanimoto coefficient is an established and generally-applicable standard, a number of viable alternatives exist, such as the Tversky coefficient. Comparatively less work has been done to extend this to graph-based similarity, though it is known that graph and fingerprint similarity largely complement each other in terms of retrieved compounds.

No one similarity method exists that is the be-all and end-all of virtual screening. This is why accompanying concepts like activity landscapes and data fusion exist, which attempt to account for shortfalls in individual methods. Group fusion in particular has been shown to be a powerful yet easily-implemented fusion method, which is also somewhat related to the hyperstructure concept (in that molecules are merged in some form).

This chapter has only discussed a small amount of the research existing in similarity. For more information on similarity, a comprehensive review by Willett (2009) details several methods and explains several concepts behind similarity searching. Willett (2013) is also a good place to further one's knowledge on data fusion.

Chapter 4

Construction and Virtual Screening of Hyperstructures

4.1 Introduction

This work represents the starting point for the three objectives outlined in the introduction. A method for constructing hyperstructures is described, along with a search procedure and a benchmark method against which the hyperstructure search is compared, using various evaluation measures. The benefits and problems of hyperstructure searching will then be assessed and summarised, with suggestions for future methodology and improvements to the concept. This chapter has an emphasis on virtual screening performance, that is - recall and diversity. Recall in this case refers to the ability of a method to prioritise ranking of active compounds over inactive ones. Diversity will be explained in more detail in the methods section.

4.2 Materials and Methods

4.2.1 Hardware and Software

The hardware used in this study featured an Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz processor with 16 GB of DDR3 RAM clocked at 1333 MHz, running Kubuntu 13.10. The Konstanz Information Miner (KNIME) 2.8.2 (Berthold et al., 2008, 2009) running Java 1.6 was used for all experimental aspects in this study, and the Chemistry Development Kit 1.5.3 (CDK) was used for all chemoinformatics functionality, unless otherwise noted (Steinbeck et al., 2003). 64-bit R 2.15.2 *R: A Language*

and Environment for Statistical Computing (2008) was used to calculate all correlation statistics reported, and the hyperstructure construction and search software was developed in Java, for use with KNIME.

4.2.2 Compound Data Sets

Three datasets have been used for the purposes of this study: the MDL Drug Data Report (MDDR) (*BIOVIA Databases | Sourcing Databases: BIOVIA Available Chemicals Directory (ACD)*, n.d.); the World of Molecular BioAcTivity (WOMBAT) (Good & Oprea, 2008); and the Maximum Unbiased Validation dataset (MUV) (Rohrer & Baumann, 2009). The MDDR and WOMBAT data sets have been used in much previous work, both at Sheffield and (in the case of MDDR) elsewhere. The MDDR data set contains 102 540 compounds, whereas WOMBAT contains 128 049 compounds. The version of MDDR used here was provided to the Sheffield chemoinformatics group for academic research purposes in late 1997. 11 MDDR activity classes were used to define the active models; the classes originally being suggested by Hert et al. (2004), giving a total of 8184 unique active molecules. Of these actives, 8099 were only active against one of the 11 classes, the rest being active against two or more targets. The version of WOMBAT used in this study was originally derived from the study of Arif, Holliday, and Willett (2009). The same processes for MDDR were also applied to 14 classes used in WOMBAT, yielding 8767 unique actives. WOMBAT actives were used as defined by Arif, Holliday, and Willett (2009), where the 14 classes in question consisted of the largest number of compounds possessing a $pIC_{50} \geq 5.0$. The MUV dataset was included as an alternative benchmark, due to its design consideration of distance equivalence. Active molecules in MUV have been selected in the datasets which are of similar structural resemblance to their chosen decoy molecules, thus avoiding analogue bias that is often found in other datasets (as in the other two data sets used here). The numbers of actives per class, with overall similarity statistics, are reported in Tables 4.1, 4.2 and 4.3 for the MDDR, WOMBAT and MUV datasets respectively. The numbers of actives for each MUV dataset have not been reported as each class in the MUV dataset contains 30 actives and 15 000 decoys.

All explicit hydrogen atoms were converted to implicit hydrogens for the purposes of this study. For each class, 10 maximally diverse compounds were selected as a training set using the MaxMin algorithm as implemented in the KNIME version of RDKit, with a randomly assigned seed (Ashton et al., 2002). These 10 molecules would be subsequently removed from the dataset to remove bias from the virtual screening statistics. 10 compounds were chosen in this case, for two reasons. Firstly,

targetID	class	Actives	\bar{S}_{AA}	S_{AA}^{σ}	\bar{S}_{AI}	S_{AI}^{σ}
6233	5HT3 Antagonist	752	0.145	0.022	0.103	0.010
6235	5HT1A Agonist	827	0.138	0.018	0.102	0.011
6245	5HT Reuptake Inhibitor	359	0.130	0.023	0.098	0.012
7701	D2 Antagonist	395	0.147	0.021	0.105	0.009
31420	Renin Inhibitor	1,130	0.288	0.051	0.114	0.006
31432	Angiotensin II AT1 Antagonist	943	0.249	0.047	0.108	0.008
37110	Thrombin Inhibitor	803	0.177	0.033	0.111	0.009
42731	Substance P Antagonist	1,246	0.153	0.019	0.109	0.009
71523	HIV-1 Protease Inhibitor	750	0.195	0.037	0.111	0.007
78331	Cyclooxygenase Inhibitor	636	0.111	0.013	0.095	0.012
78374	Protein Kinase C Inhibitor	453	0.124	0.029	0.097	0.014

Table 4.1

The numbers of actives and similarity statistics for the classes used in the MDDR dataset, using RDKit standard Morgan fingerprints at a radius of 3 as the descriptor. \bar{S} represents the mean pair-wise similarity (MPS) and S^{σ} is the standard deviation of similarity values. Subscript AA denotes similarity between active molecules in the class, and AI is the inter-similarity between actives and inactives.

targetID	class	Actives	\bar{S}_{AA}	S_{AA}^{σ}	\bar{S}_{AI}	S_{AI}^{σ}
1	5HT3 Antagonist (rat)	474	0.319	0.129	0.118	0.039
2	5HT1A Agonist (rat)	503	0.142	0.102	0.104	0.032
3	D2 Antagonist (rat)	1,128	0.193	0.104	0.111	0.036
4	Renin Inhibitor (human)	421	0.206	0.119	0.115	0.037
5	Angiotensin II AT1 Antagonist (rat)	722	0.240	0.113	0.110	0.032
6	Thrombin Inhibitor (human)	220	0.173	0.127	0.095	0.030
7	Substance P Antagonist (human)	142	0.323	0.201	0.095	0.030
8	HIV-1 Protease Inhibitor (human)	596	0.146	0.105	0.102	0.030
9	Cyclooxygenase Inhibitor (human)	558	0.176	0.101	0.109	0.035
10	Protein Kinase C Inhibitor (rat)	842	0.174	0.111	0.109	0.032
11	Acetylcholine esterase inhibitor (human)	910	0.159	0.090	0.104	0.034
12	Factor Xa inhibitor (human)	694	0.188	0.102	0.109	0.035
13	Matrix metalloprotease inhibitor (human)	965	0.156	0.100	0.091	0.030
14	Phosphodiesterase inhibitor (human)	592	0.189	0.116	0.106	0.034

Table 4.2

The numbers of actives and similarity statistics for the classes used in the WOMBAT dataset.

this number was chosen to leave enough active compounds available for searching for in each dataset. Notably, this was of particular concern with the MUV dataset - as most classes have only 30 actives, removing the 10 compounds would leave 20 actives in the dataset to search for. Secondly, the findings of Klinger and Austin (2006) found that the benefit from adding more reference compounds was

targetID	class	\bar{S}_{AA}	S_{AA}^{σ}	\bar{S}_{AI}	S_{AI}^{σ}
466	sphingosine-1-phosphate 1 receptor potentiators	0.121	0.054	0.117	0.037
548	protein kinase A inhibitors	0.134	0.083	0.105	0.037
600	steroidogenic factor 1 inhibitors	0.126	0.063	0.118	0.038
644	rho kinase 2 inhibitors	0.123	0.074	0.106	0.039
652	HIV reverse transcriptase RNase	0.108	0.049	0.109	0.038
689	ephrin type-A receptor 4 antagonist inhibitors	0.114	0.043	0.110	0.034
692	steroidogenic factor 1 activators	0.113	0.039	0.113	0.037
712	heat shock protein 90 kDa alpha inhibitors	0.108	0.047	0.105	0.033
713	estrogen receptor-alpha coactivator binding inhibitors	0.116	0.051	0.115	0.036
733	estrogen receptor-beta coactivator binding inhibitors	0.114	0.053	0.112	0.038
737	estrogen receptor-alpha coactivator binding potentiators	0.131	0.051	0.120	0.039
810	focal adhesion kinase inhibitors	0.112	0.059	0.106	0.037
832	Cathepsin G	0.154	0.097	0.120	0.035
846	factor XIa	0.162	0.092	0.116	0.037
852	factor XIIa	0.152	0.092	0.115	0.034
858	dopamine receptor D1 allosteric modulators	0.116	0.060	0.111	0.037
859	muscarinic receptor M1 allosteric modulators	0.126	0.037	0.121	0.036

Table 4.3

The numbers of actives and similarity statistics for the classes used in the MUV datasets.

increasingly diminishing in virtual screening (up to 5 references). 10 compounds was chosen as the MCS algorithm used was fast enough to handle this alignment, whilst not having a detrimental effect on the searching of any particular dataset - both in terms of actives per dataset, and virtual screening power of the hyperstructure. RDKit standard Morgan fingerprints were used with a maximum radius of 3 (similar to the ECFP_6 fingerprint found in Pipeline Pilot (Rogers & Hahn, 2010), folded into 1024 bits) as the fingerprint descriptor for this study. RDKit standard Morgan Fingerprints differ only by the atom typing definitions to ECFP fingerprints, in that isotope information is added, and the valance-hydrogen count parameter is removed (Landrum, 2012). ECFP_4 and ECFP_6 are both fingerprints that are known to achieve high recall rates, though ECFP_6 has been shown to be slightly more powerful in similarity studies (Papadatos et al., 2009).

4.2.3 Maximum Common Substructure

The MCES for all purposes of this chapter, was found using the MaxCommonSubstructure method developed for JChem 6.1.0, 2014 by ChemAxon (Englert & Kovács, 2015; Kovács & Englert, 2013). This method was chosen with speed in mind, instead of the size of the common substructure retrieved (though we found the latter to be satisfactory in internal tests). Additional performance information of

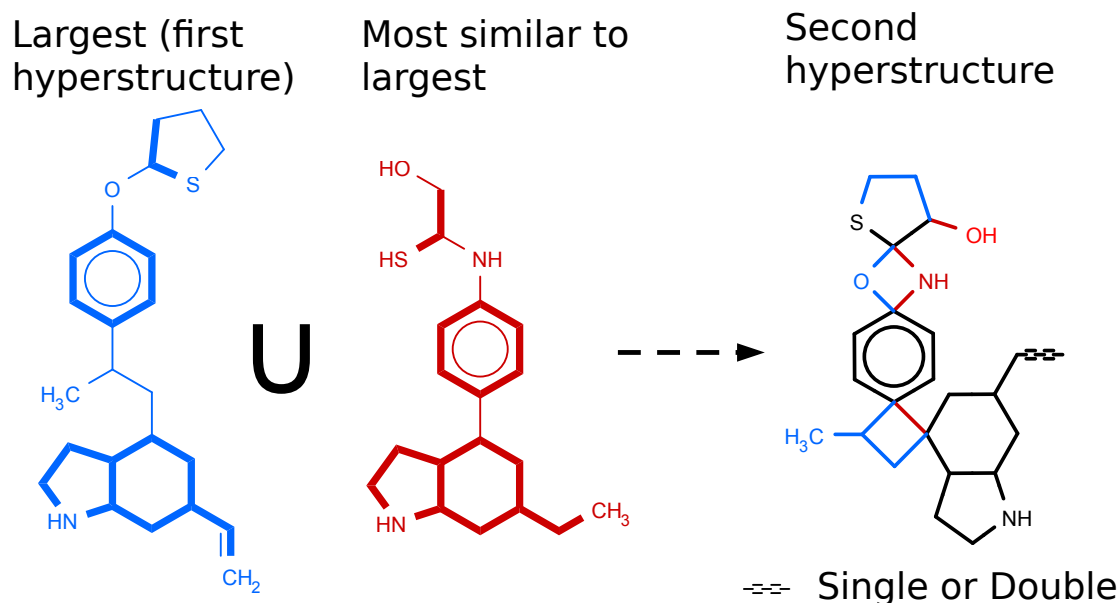


Figure 4.1. The hyperstructure construction process with two molecules, shown in blue and red - bold bonds indicate those in the MCS between the two molecules. In the resulting hyperstructure, black bonds and atoms indicate the MCS, while unique bonds and atoms are coloured based on the original molecules.

this algorithm can be found in Chapter 6, and background information on the algorithm is in Chapter 2.

4.2.4 Hyperstructure Construction and Application

Construction in this methodology is based on the method of (R. D. Brown, Downs, et al., 1994). The construction process (Figure 4.1) used is as follows, starting from a set of input molecules:

1. Select the largest molecule and remove it from the set of available molecules. This molecule is now the first hyperstructure.
2. Select the next most similar molecule, based on the number of bonds in common between the hyperstructure and this molecule.
3. Use the MCS procedure to overlap and then to append this molecule to the hyperstructure, and remove this molecule from the set. Bonds of different types may be overlapped, yielding degenerate bond types (depicted by dashed lines in the hyperstructures).
4. Repeat steps 2 and 3 until the set is empty.

This process relied on finding the MCES between two molecules to produce the hyperstructures.

Algorithms 3 and 4 (in Appendix A) detail the process for which the MCES is used. Briefly, bonds in the non-hyperstructure were tested for presence or absence in the MCES. Bonds absent from the MCS were added to the hyperstructure (and any “missing” atoms as well), whereas bonds in the hyperstructure that are in the MCES, were adjusted to allow for both the relevant hyperstructure and non-hyperstructure bond to match. The ChemAxon MCS was obtained using default settings, except that bond types were ignored. In the case where bonds were matched with different labels, the resulting bond in the hyperstructure was adjusted to allow mapping of all bond types in subsequent searches. If aromatic and single bonds overlapped, then the resulting hyperstructure bond created would allow the mapping of both single and aromatic bonds. Although different bond labels were allowed to map, for this work we maintained the rule that identical atom labels must match. Whilst it would be interesting to see what effect this has, it is beyond the scope of this work.

4.2.5 Similarity Searching

The Tanimoto coefficient is the standard similarity coefficient for quantifying chemical structural similarity. For standard MCS searching in this study, the Tanimoto coefficient has been used. When concerning hyperstructures however, we decided to use the asymmetric Tversky coefficient over symmetric coefficients. Several advantages of this coefficient in virtual screening are mentioned in Chapter 3. We believe that this coefficient is more suitable for similarity searching using hyperstructures due to the size differences (notably in this work, the number of bonds) between hyperstructure and database molecules. The hyperstructure is highly likely to be significantly bigger than the majority of database molecules, let alone the molecules used to construct the hyperstructure. In addition, it is expected that substructure similarity plays a large part in suitable similarity searching, as the hyperstructure collectively represents the scaffolds present in the input molecules. The Tversky coefficient can be biased towards substructure similarity, in order to take advantage of this hypothesis.

The Tversky coefficient is described in Chapter 3. In this chapter, c represents the number of common edges between the hyperstructure and the database molecule, as calculated using the JChem dMCES method. The hyperstructure has a bonds, and the database molecule has b bonds. β was set to 0.3; 0.5 (symmetric); 0.7; 0.8; 0.9; 0.95; and 1.0 to determine how this variable had an affect on similarity searching. Higher values of β give a near substructure-like search, whereas lower values bias the results towards a superstructure-like search. It is expected that higher values of β will yield better retrieval rates as the size difference between hyperstructure and database molecule is rendered less important.

As a comparison to gain some idea of the hyperstructure's performance in similarity searching, two additional search methods were employed. The methods were as follows:

- Hyperstructure construction and searching, using the Tversky coefficient with the aforementioned values of β
- MCS group fusion using the Tanimoto coefficient (either using MAX or AVG fusion on the similarity scores)
- RDKit Morgan fingerprint group fusion at a maximum radius of 3 for the fingerprints. Both MAX or AVG fusion is applied on the similarity scores separately.

The MAX fusion rule has generally been shown to be the most suitable fusion rule in group fusion virtual screening (Whittle et al., 2004), and is expected to thus outperform the AVG rule in this study. Although somewhat superior fusion rules exist (refer to Chapter 3), this study is not concerned with seeking the best fusion rule - the rules here have simply been used as effective benchmarks.

4.2.5.1 Similarity Fusion. As it is known that graph theory similarity and fingerprint similarity are somewhat complementary (and thus the same is assumed of hyperstructure searches) (Raymond & Willett, 2002a), it is assumed that the three methods would be suitable candidates for further data fusion. AVG fusion has thus been applied to the three rules to give three pairwise combinations, as well as the AVG of all 3. The AVG rule has been shown in general to be more suited to data fusion of ranks, than the MAX fusion rule (Willett, 2013), noting that when concerning ranks and similarity scores, SUM and AVG rules yield identical orders. In this case, the hyperstructure searches have been performed with a β of 0.9 as this was found to be the optimal value for fruitful virtual screening from a number of sources (Daylight Chemical Information Systems, 2011; Horvath et al., 2013). In addition, as explained in the results section, this was found to be a generally satisfactory value for suitable recall rates in our experiments. The fusion methods are summarised in Figure 4.2. Table 4.4 also explains the meanings of the method abbreviations used here.

4.2.6 Evaluation Metrics

4.2.6.1 Virtual Screening. The Boltzmann-Enhanced ROC score (BEDROC) has been chosen to evaluate the power of the similarity searches. A ROC (ROC) curve is a plot of the true positive rate against the false positive rate, and can be used to determine how well a model classifies data. The

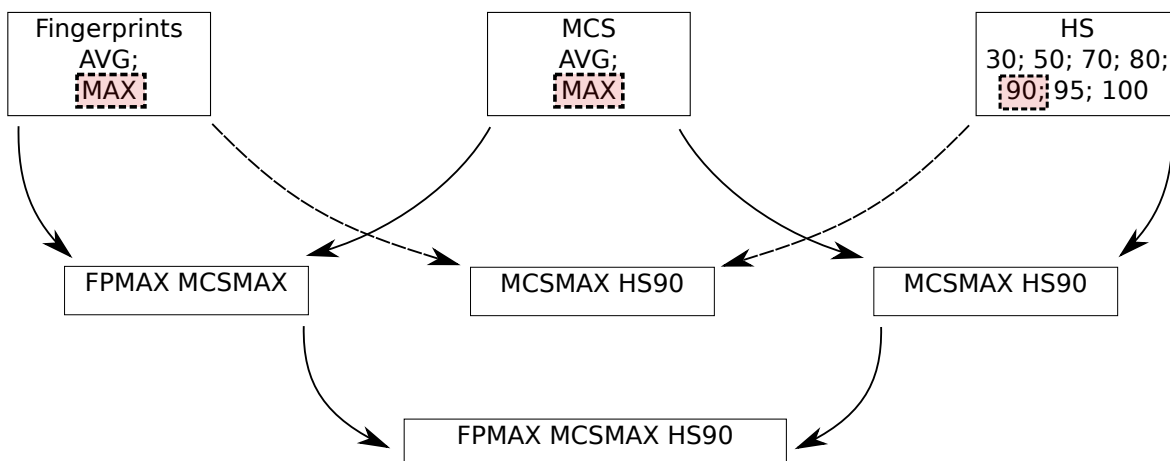


Figure 4.2. Summary of the search methods employed in this study. The arrows indicate which search methods are fused via the MAX rule (top-down)

Method	Description
(Tversky) HS 30	Hyperstructure search applied using Tversky similarity coefficient, with a β of 0.3
(Tversky) HS 50	As above, with a β of 0.5
(Tversky) HS 70	As above, with a β of 0.7
(Tversky) HS 80	As above, with a β of 0.8
(Tversky) HS 90	As above, with a β of 0.9
(Tversky) HS 95	As above, with a β of 0.95
(Tversky) HS 100	As above, with a β of 1.0
FP MAX	Fingerprint Tanimoto similarity with the MAX fusion rule applied to the similarity scores
FP AVG	Fingerprint Tanimoto similarity with the AVG fusion rule applied to the similarity scores
HS MAX	The best reported hyperstructure similarity, out of all the values of β used
MCS MAX	Maximum Common Substructure Tanimoto similarity (based on the bonds in the MCS and the two structures being compared), with the MAX fusion rule applied to the similarity scores
MCS AVG	As above, but using the AVG fusion rule
FPMAX MCSMAX	SUM fusion of FP MAX and MCS MAX ranks (not scores)
FPMAX HS90	SUM fusion of FP MAX and Tversky HS 90 ranks
MCSMAX HS90	SUM fusion of MCS MAX and Tversky HS 90 ranks
FPMAX MCSMAX HS90	SUM fusion of FP MAX, MCS MAX and Tversky HS 90 ranks

Table 4.4

Descriptions of the abbreviations of search methods employed here.

area under this curve (AUROC) ideally should be as close to 1.0 as possible (better at classifying), as a value of 0.5 indicates that the model is no better than random. BEDROC in this case represents

a special case of ROC in that it ranks a method on its ability to retrieve active compounds as early in the ranking order as possible, rather than its overall classification accuracy, which is what AUROC denotes. Thus, BEDROC is better suited to assessing virtual screening ability than AUROC is. Another advantage of BEDROC is that it is not as easily influenced as AUROC by the ratio of actives to inactive compounds (or positive to negative data). Like AUROC, BEDROC scales from 0.0 to 1.0, a value closer to 1.0 indicating superior virtual screening performance whereas a value closer to 0.0 indicates the worst virtual screening performance (random). Unlike AUROC, however, a BEDROC of 0.5 implies intermediate virtual screening ability, instead of random, as it would for AUROC (Truchon & Bayly, 2007).

A small problem with BEDROC is the requirement of setting a tuning factor α , which determines how many of the top-ranked compounds contribute to the BEDROC score. A higher value of α means that a smaller percentage of the top-ranked compounds contribute to the majority of the BEDROC score. A value of 160.9 for α has been chosen in this study as it corresponds to an Enrichment factor (EF) of 1%, where 80% of the BEDROC score is explained by the top 1% of compounds in the ranked list (Truchon & Bayly, 2007). The EF for the top 1% of ranked compounds was also included as it is easier to interpret than the BEDROC score. It should be noted that whilst it has generally been shown that BEDROC and EF are highly correlated (Riniker & Landrum, 2013), BEDROC takes account of the ratio of actives to inactives, where EF does not. The same study notes that the two measures are uncorrelated when this ratio differs between activity classes.

4.2.6.2 Hyperstructure Information. Compression (of bonds) in this case, is defined by the equation

$$C_B = \frac{\sum_1^{N_Q} B_{Qi}}{B_H \times N_Q}$$

where B_{Qi} is the number of bonds in query structure i , B_H is the number of bonds in the resulting hyperstructure, and N_Q is the number of query structures used to build the hyperstructure. This is a standardised measure resulting in a value from 0 to 1, where a value closer to 1 indicates a superior compression. This statistic can be thought of as the proportion of bonds from the queries that have been fully overlapped in the hyperstructure. The same analogy is made with atoms, and is referred to here as C_A . It is expected generally that compression statistics correlate with the average similarity of the active compounds in the class, though due to the diverse nature (in both structure and size) of

active compounds selected, this is not expected to be a strong correlation.

4.2.6.3 Diversity Evaluation. Although it is useful to observe the overall virtual screening performance of a method, it is also of interest to see how powerful a method is in obtaining a diverse and heterogeneous set of compounds. In some cases, the diversity is actually more important than simply the active compound retrieval if it leads to finding a new scaffold for a particular therapeutic problem domain. A “scaffold” in this work has been defined as a Bemis-Murcko framework (Bemis & Murcko, 1996), with atom and bond information removed, and has been calculated using the RDKit tools in KNIME. We have utilised a number of statistics in this work to measure diversity.

The first statistic is the “First-Found” scaffold enrichment factor, using the top 1% of the ranked list of molecules (represented here as $EF_{FF1\%}$). “First-Found” refers to the rank of the top-ranked molecule belonging to a scaffold, thus ignoring all subsequent molecules belonging to the scaffold. Although this measure of obtaining diversity has been criticised for several statistical flaws (Mackey & Melville, 2009), we have used it here as we are only interested in whether an active scaffold is identified or not in a ranked list. We are not interested in how the compound ranks are distributed for the given active scaffold. This represents a common goal in a virtual screening project, if one is seeking new scaffolds.

The second measure is the mean number of active molecules per active scaffold in the top 1% ranking (represented here as $\bar{x}_{F1\%}$). A lower value of this statistic indicates that a search method retrieves on average fewer actives per scaffold, and is therefore less biased towards finding analogues for a small number of scaffolds. The standard deviation ($s_{F1\%}$) of the number of actives per active framework has also been calculated.

The mean pairwise similarity (using the same fingerprints as for searching) of said actives in the top 1% ranking has also been calculated to complement the framework statistics, represented as $\bar{S}_{F1\%}$. We regard this statistic to be less useful than framework information as it is less able to distinguish between a set of moderately similar analogues of one series, compared to several series with very similar analogues.

4.2.6.4 Complementarity. Another aspect of interest is to assess the complementarity of the compounds retrieved for two methods, especially when comparing the fingerprinting similarity with hyperstructure similarity. This can be of potential interest in the aspect of similarity fusion - where rankings from different similarity search methods are fused to obtain a consensus score. To compare rank lists, the top 5% of ranked compounds were used, and the number of matched pairs were

Method	Description
\bar{S}_{AA}	MPS between active compounds in an activity class, of a mentioned dataset
S_{AA}^{σ}	Standard deviation of the above
\bar{S}_{AI}	MPS of actives compared with inactives in an activity class, of a mentioned dataset
S_{AI}^{σ}	Standard deviation of the above
a	The number of on bits (or edges) in the reference molecule of the similarity coefficients used in this article
b	The number of on bits (or edges) in the database molecule
c	The number of on bits in common between the reference and database molecule. Alternatively when concerning graph theory - the number of edges in the MCS
β	The weighting used in the Tversky coefficient to bias towards either substructure and superstructure similarity
α	parameter which determines how many of the top-ranked compounds contribute to the BEDROC score. Note that this is unrelated to β !
C_B	Bond compression of a hyperstructure
C_A	Atom compression of a hyperstructure
$EF_{FF1\%}$	“First-Found” scaffold enrichment factor, using the top 1% of the ranked list of molecules.
\bar{x}_{F1}	The mean number of active molecules per active framework, in the top 1% of compounds identified.
s_{F1}	standard deviation of the above
\bar{S}_1	mean pairwise similarity of the top 1% of compounds identified, using Morgan fingerprints with a radius of 6 and the Tanimoto coefficient.

Table 4.5

Descriptions of symbols which are frequently referred to in this chapter.

recorded per comparison (a matched pair meaning that the same compound occurs in both lists). These comparisons are represented as heatmaps in Appendix C. R was used to compute this value for each dataset, as well as performing Wilcoxon signed-rank tests and obtaining the matched pair numbers.

4.2.6.5 Physicochemical Properties. To get an idea of whether the search techniques retrieved different sets of molecules, we calculated a number of physicochemical descriptors. These were calculated for the active compounds in the top 1% of the ranked database resulting from each similarity search method. The descriptors used were the counts of the number of rotatable bonds, heavy atoms, heteroatoms and rings; the length of the largest acyclic chain, and the fragment complexity. This last

descriptor was the CDK implementation of the work described by Nilakantan et al. (2006):

$$C = |B^2 - A^2 + A| + H/100$$

where A is the number of non-hydrogen atoms, B is the number of bonds, C is the fragment complexity and H is the number of heteroatoms.

To assess the overlap between the ranked lists resulting from two different search methods, we calculated the number of actives common to the top 1% of the two ranked lists.

4.3 Results and Discussion

4.3.1 Virtual Screening Recall Performance

Figures 4.3, 4.4, 4.5 displays the BEDROC scores for each activity class, for the similarity search methods mentioned, in addition to HS 80 and HS 95, and the four SUM fusion methods as depicted in Table 4.4. Figure 4.6 (and Table E.1 in Appendix E) displays summary virtual screening statistics for FP MAX, (Tversky) HS 90, MCS MAX, and the fusions of the three techniques (as detailed in Table 4.4). Figure 4.6 displays $EF_{1\%}$ and BEDROC for each method, mean-averaged over all the activity classes for a dataset. The immediate conclusion from these statistics is, on comparing the mean values of BEDROC score, that the FP MAX was significantly superior to HS MAX, with MCS MAX lying between the two. In all three datasets the fingerprints significantly (and consistently) outperformed hyperstructures in virtual screening ability, as seen from the mentioned Table and Figures. This suggests that fingerprints were better suited to virtual screening (on these datasets), the p-values for the Wilcoxon signed-rank tests being significant (Figure 4.6). As seen in Figures 4.3, 4.4, 4.5 however, MCS MAX in some of the classes (notably where the BEDROC scores were lower) outperformed FP MAX, though as FP MAX's performance improved, MCS MAX improved at a slower pace and was quickly outperformed. The observation that fingerprints generally match or outperform MCS-based methods is consistent with the related work of Raymond and Willett (2002a), though in this work the fingerprints used are of a better standard in terms of virtual screening recall.

Virtual screening statistics for each activity class are tabulated in Appendix D. A noteworthy observation is the contrast between the EF and BEDROC scores. Using Table D.6 as an example, two values of 20 have been achieved for the EF, yet the BEDROC score for Substance P Antagonists was higher than for 5HT3 Antagonists. As the actives clustered at superior ranks in Substance P Antagonists than for 5HT3 antagonists, this implies that the ranking of the activity class with the higher BEDROC is

better, despite the search method identifying all active compounds in the top 5% fraction for both activity classes.

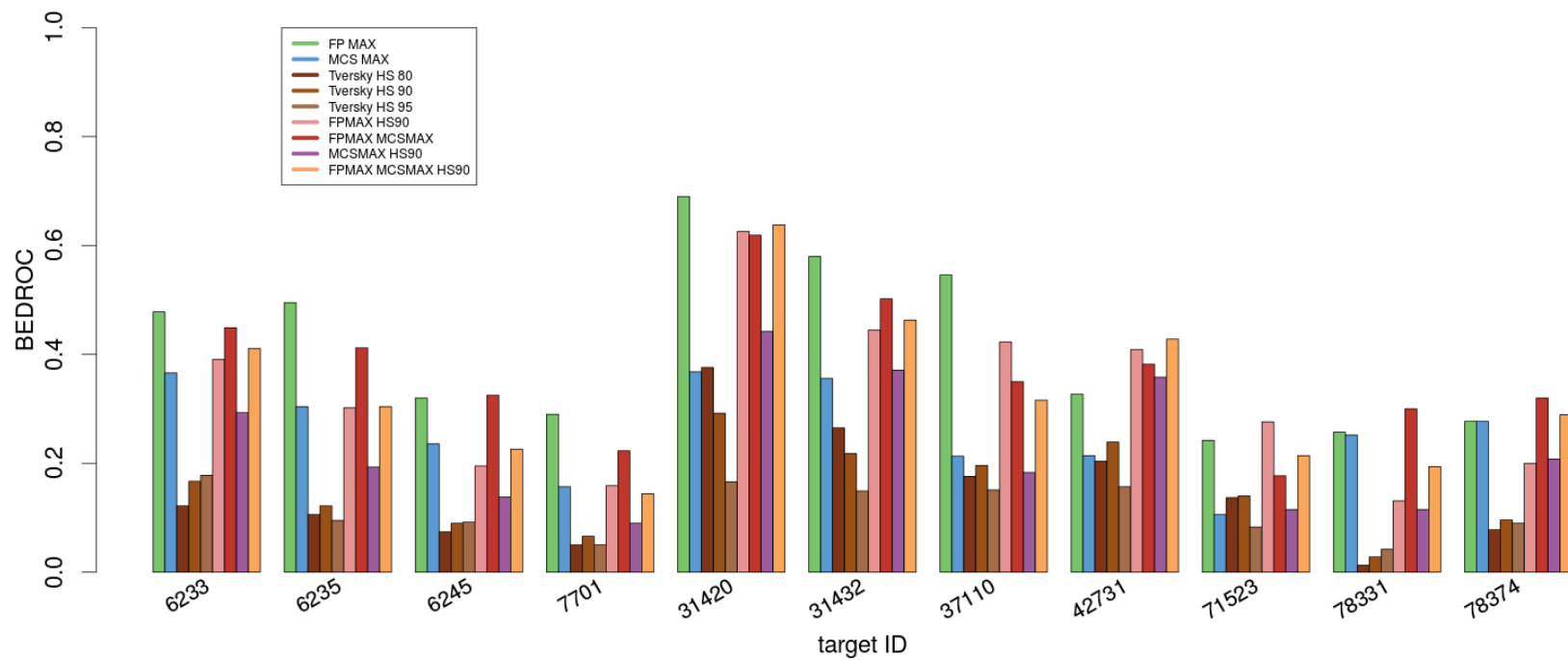


Figure 4.3. BEDROC scores for the search methods, per activity class, for MDDR

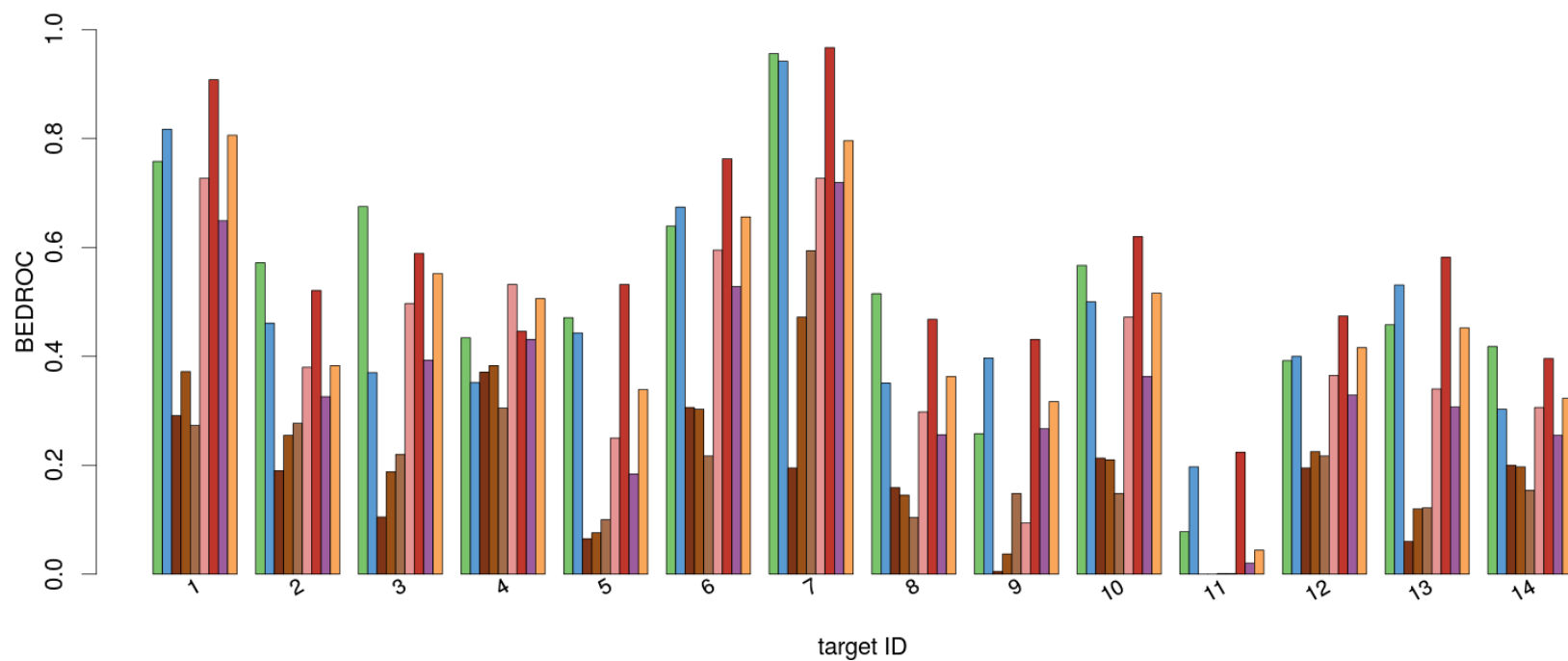


Figure 4.4. BEDROC scores for the search methods, per activity class, for WOMBAT

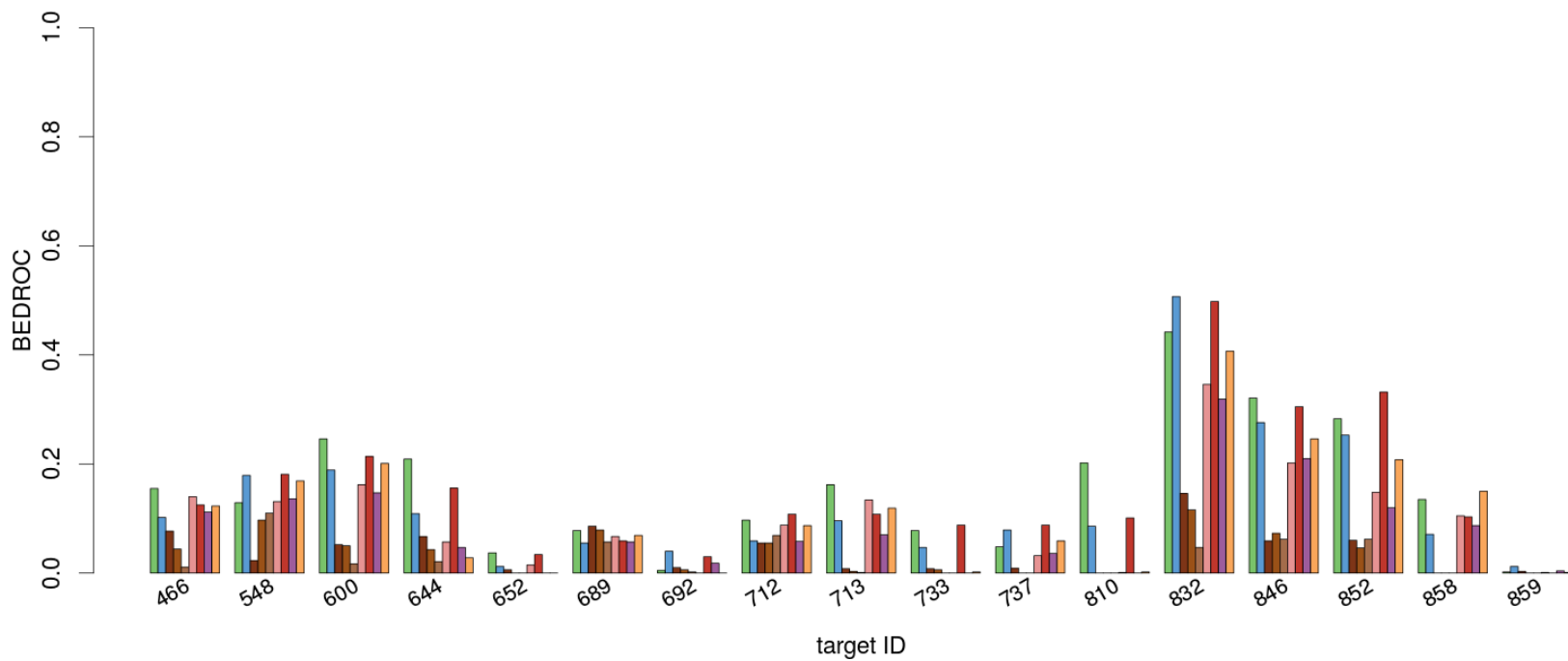


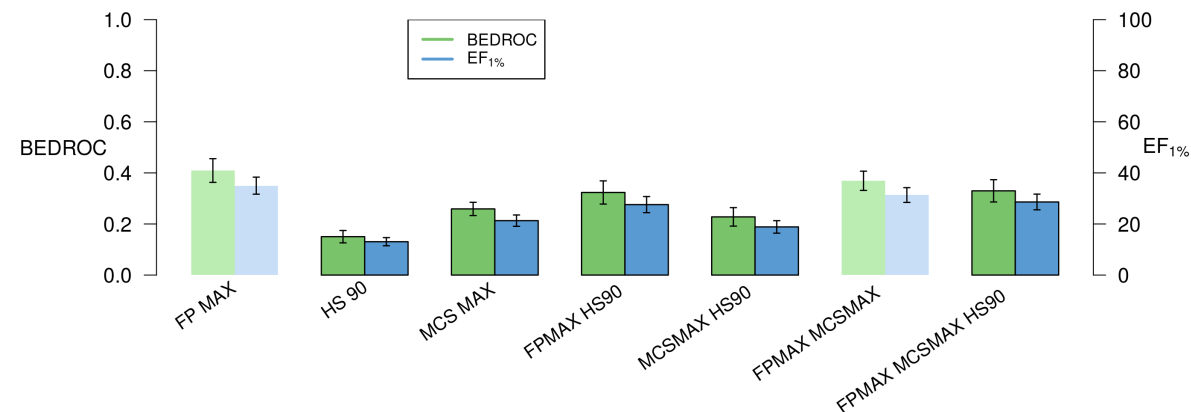
Figure 4.5. BEDROC scores for the search methods, per activity class, for MUV

In the specific case of MUV, the results are somewhat harder to interpret than the other datasets, due the statistics being based on a small number of actives (20) being sought after. Fingerprints again generally performed better than the other two methods. Of interest are the 4 cases where the best value of β was 0.5 or lower - in contrast to the other two datasets where higher values of β have been shown to be best. For these cases, an increase in the value of β actually worsened the BEDROC and EF scores (Table D.7). For the cases mentioned where the value of β was 0.3 or 0.5, these results suggest that the active molecules in the test sets had a lower similarity to the respective hyperstructures than inactives, which seems counter-intuitive. It is unclear why this may have been the case, though the comparatively unbiased nature of the MUV dataset, and the small number of active compounds may explain this.

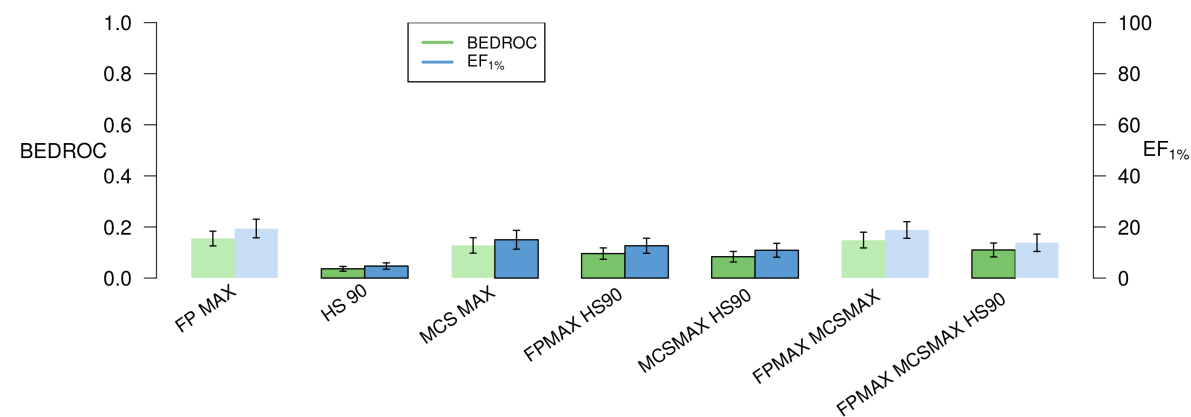
4.3.2 Diversity Performance

Figure 4.7 (and Table E.2 in Appendix E) displays summary virtual screening statistics (relating to scaffold retrieval) for FP MAX, (Tversky) HS 90, MCS MAX, and the fusions of the three techniques (as detailed in Table 4.4). The charts in Appendix B display the $EF_{FF1\%}$ and $\bar{x}_{F1\%}$ for each activity class, for the similarity search methods mentioned, in addition to HS 80 and HS 95. Figure 4.7 summarises the same information across the activity classes as mean averages.

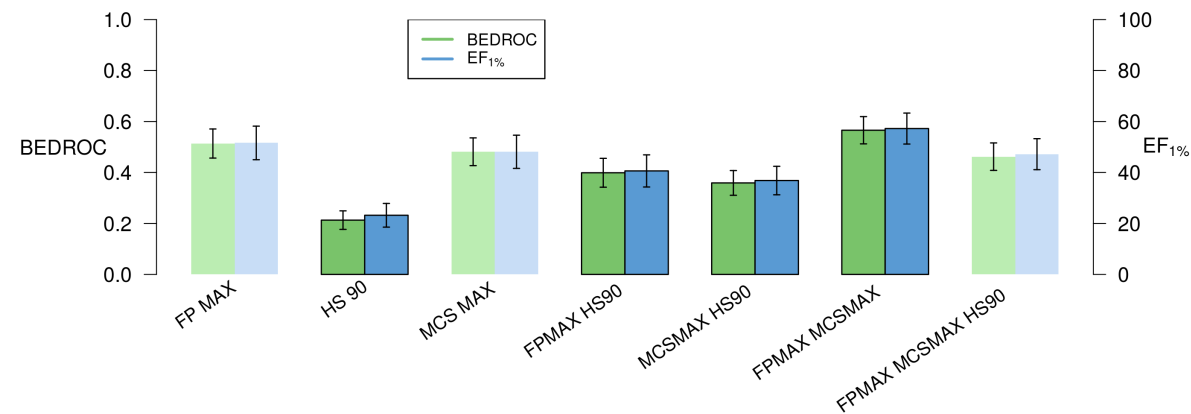
Across all classes of the three datasets, it can be seen that the hyperstructures almost always yielded a lower $\bar{x}_{F1\%}$ than the FP MAX and MCS MAX methods, the latter actually (though not statistically significant) having the largest number of compounds per framework of all methods presented. For hyperstructures, these results were statistically significant for MDDR and MUV as shown in Figure 4.7, suggesting that the top-ranked active molecules retrieved by hyperstructures were more heterogeneous than those retrieved with fingerprints (in relation to the number of actives retrieved). However, hyperstructure searches here retrieved a far smaller number of frameworks than fingerprints did (as evidenced by a significantly lower scaffold enrichment factor). This suggests that although hyperstructures retrieved a less analogous set of actives, fewer active scaffolds were retrieved overall. In addition, in Figure B.6 (MUV), points with a $\bar{x}_{F1\%}$ of 0.0 retrieved no active molecules in the top 1%, and therefore should be ignored for diversity comparisons. No apparent advantage (and for that matter, no significant difference) is seen in the mean pairwise similarity of actives retrieved across the three datasets, suggesting that hyperstructures were better at picking a greater variety of scaffolds rather than generally more dissimilar sets (Appendix D). MCS MAX by comparison had no significant difference in framework retrieval compared to fingerprints, and combined with its inferior



(a) MDDR

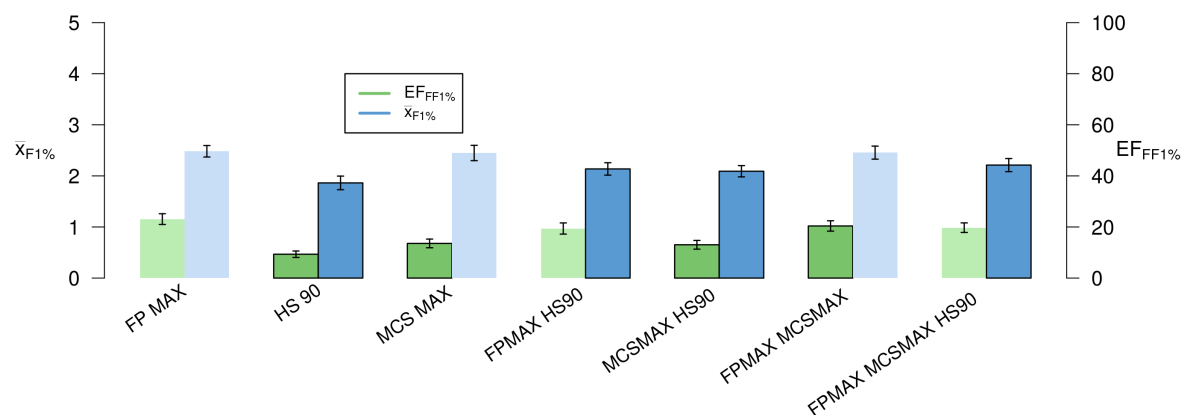


(b) MUV

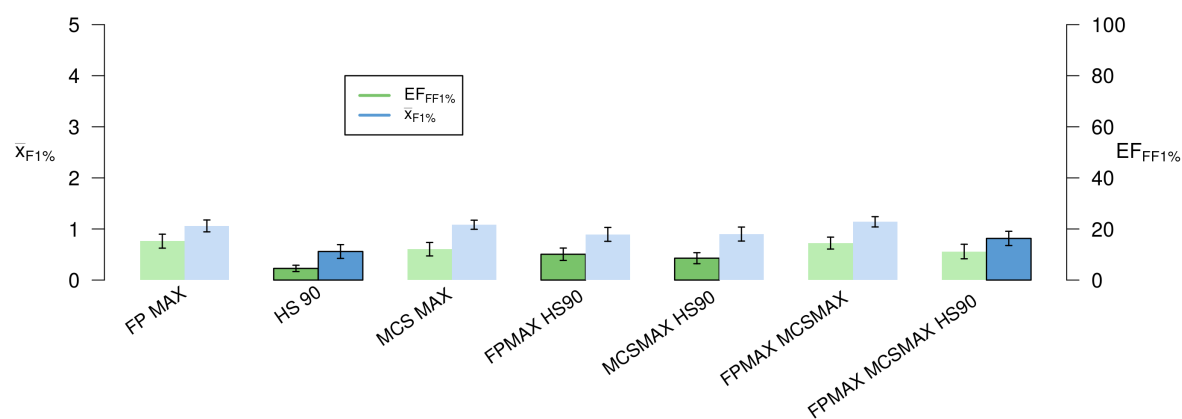


(c) WOMBAT

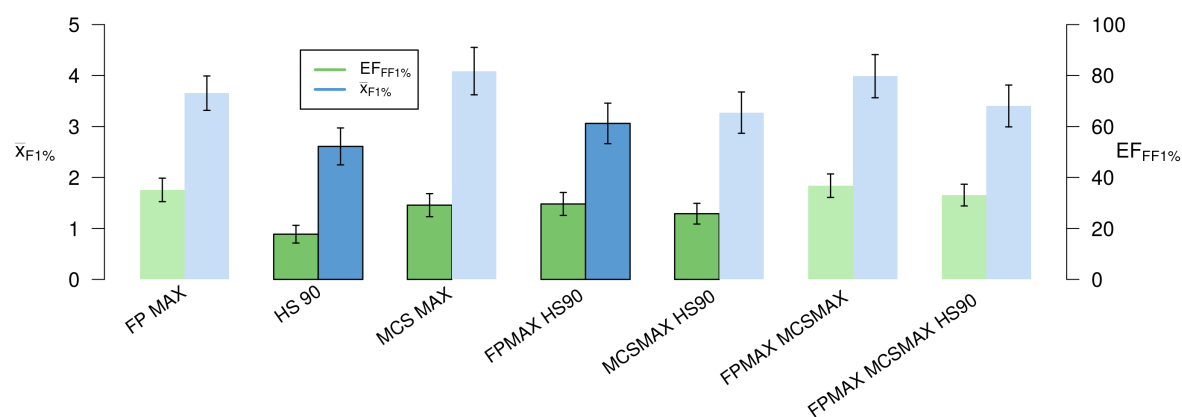
Figure 4.6. Bar charts showing mean recall statistics for the datasets. Dark-coloured bars indicate that the method has a mean significantly different from that of FP ($p \leq 0.05$), as determined by a paired 2-tailed Wilcoxon signed-rank test. Error bars represent one standard error of the mean, above and below the mean.



(a) MDDR



(b) MUV



(c) WOMBAT

Figure 4.7. Bar charts showing mean scaffold-retrieving statistics for the datasets. Dark-coloured bars indicate that the method has a mean significantly different from that of FP ($p \leq 0.05$), as determined by a paired 2-tailed Wilcoxon signed-rank test. Error bars represent one standard error of the mean, above and below the mean.

virtual screening performance (compared to fingerprints), hyperstructure searches seem more useful for retrieving a diverse set of actives, as opposed to retrieving a greater proportion of scaffolds.

4.3.3 Relative Performances

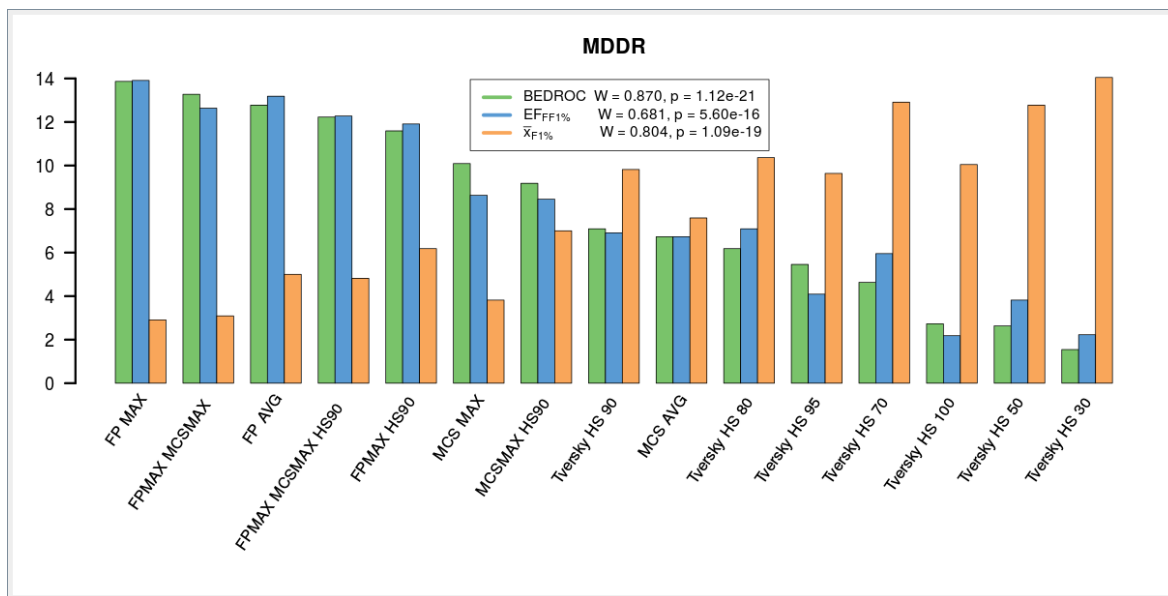


Figure 4.8. Bar chart for the MDDR dataset, showing the ranks of three parameters for each similarity method, that are being optimised for in this study. Methods are sorted in descending order of their BEDROC rank, where the higher the value of the rank - the better the property is. Values for Kendall's W and the associated p value are given for the relevant statistic for each dataset, where Kendall's W quantifies the agreement in rankings between the activity classes of the dataset for a search method.

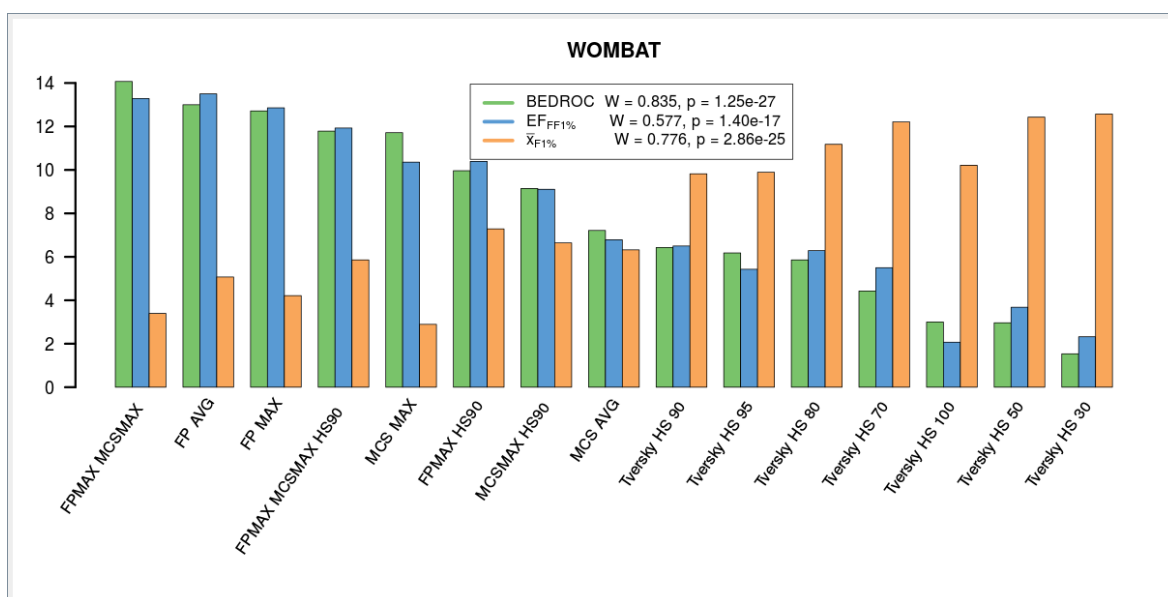


Figure 4.9. Bar as described in Figure 4.8, but for the WOMBAT dataset

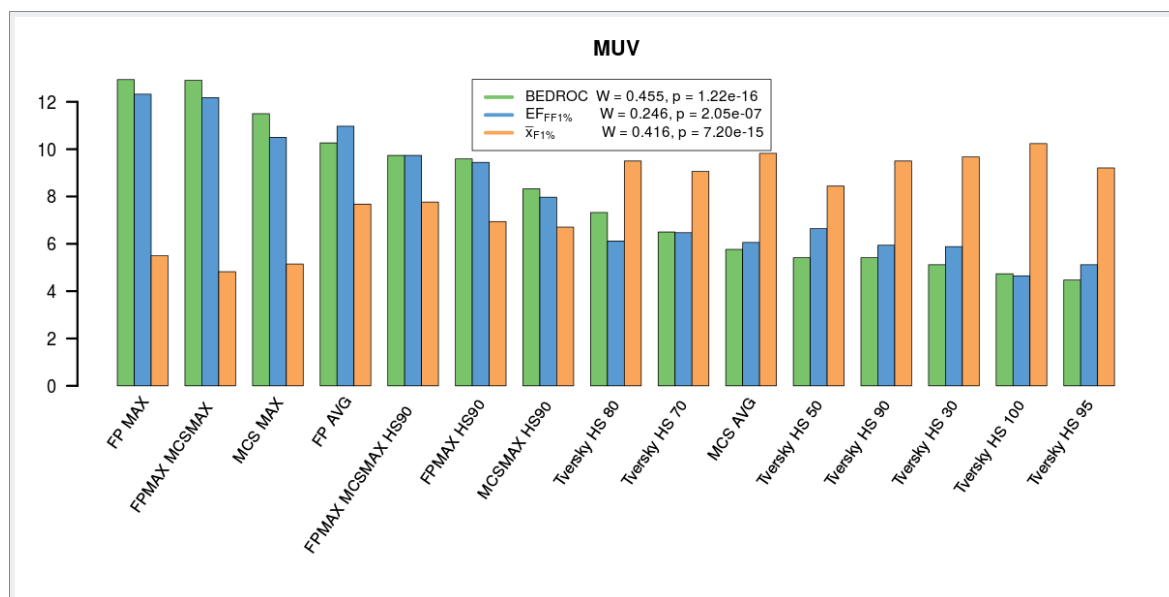


Figure 4.10. Bar as described in Figure 4.8, but for the MUV dataset

The value of β appears to have had a strong influence on the recall statistics, and to a lesser extent the diversity of the actives retrieved. Kendall's W test was applied to the BEDROC scores to attempt to find consistent agreement between the different activity classes, for a given search method. The results are presented in Figures 4.8, 4.9 and 4.10 (the values of the mean ranks are in Table F.1), where it is clear that all the rankings are significant - that is, for each activity class, a method ranks similarly, thus implying a consistent performance of a search method. β values of 0.8, 0.9 and 0.95 generally rank highest for hyperstructure similarity searches, somewhat confirming what we expected (as mentioned in the Methods section). The same analysis using Kendall's W test was performed using $EF_{1\%}$, though as the results were almost identical to the BEDROC rank results, they have not been shown here.

Kendall's W test was also performed on the two framework statistics (Mean rank values shown in Tables F.2 and F.3) There is less agreement than with the BEDROC scores, especially in WOMBAT (though the WOMBAT W value is still significant). β values of 0.8 and 0.9 performed better than other values for scaffold enrichment (Table F.2). Otherwise, the trends observed here were relatively similar to those for the BEDROC ranks.

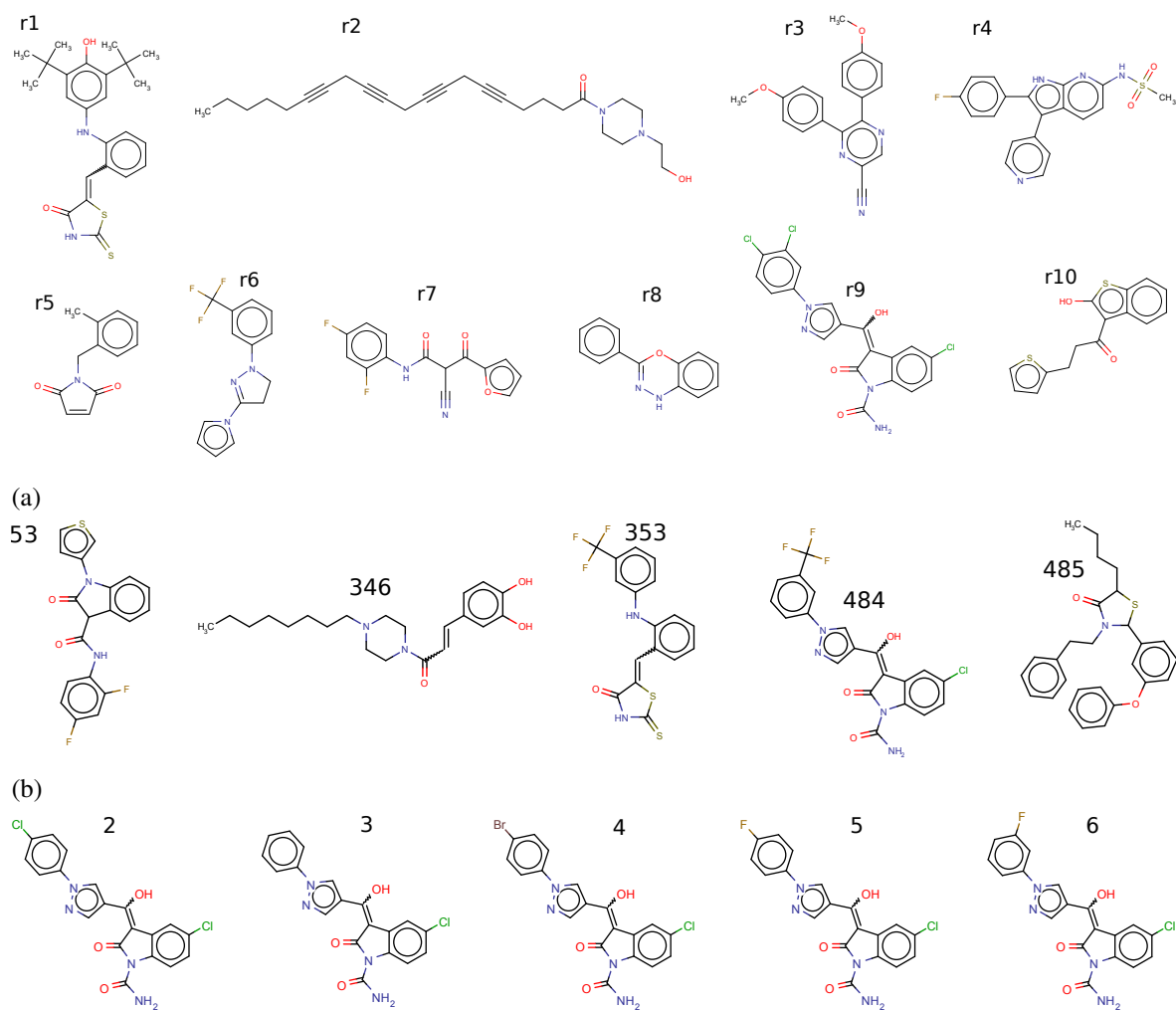
When concerning $\bar{x}_{F1\%}$, 0.9, the β value that scored best for BEDROC was actually one of the worst values for obtaining diverse results, at least in the context of hyperstructures. In fact, it appears that diversity and BEDROC were opposed for all the methods - a result not entirely surprising, considering that active compounds which are more similar to inactives than the actives, are more likely to be

chosen by less powerful methods. MCS MAX and FP MAX are extreme examples - techniques with relatively high BEDROC scores, yet they retrieved several compounds per unique framework. This implies that higher performing techniques exploit the similar property principle better, retrieving a greater proportion of analogues. Likewise, this also implies that hyperstructures are actually not so suitable for analogue retrieval. It is evident that essential information relating to compound activity is lost when constructing hyperstructures in this work, as individual queries (through MAX fusion) outperformed hyperstructures. This is particularly evident in the more heterogeneous activity classes, like cyclooxygenase inhibitors.

To show the differences in the kinds of active compounds retrieved by hyperstructures and fingerprints, the top five ranked actives involved in the MDDR COX inhibitors search are shown in Figure 4.11b, with the reference structures from which the hyperstructure was constructed being in Figure 4.11a. Comparisons of these two reveal that the hyperstructure similarity search identified three scaffolds not present in the reference structures (actives 56, 346 and 485). Of note, the top five compounds retrieved by FP MAX (Figure 4.11c) all share the same scaffold (with r9) and generally differ from each other by just one atom. This ability to prioritise analogues over non-analogues is a major reason for why FP MAX outperformed Tversky HS 90 (and it is unsurprising this works so well given that 10 molecules with different scaffolds are used as the reference set).

4.3.4 Method Complementarity

The overlaps of the ranks of the matched compounds are summarised in the heatmaps in Figure C.1 in Appendix C. As most of the heatmaps display similar results, one example has been picked and displayed in Figure 4.12. The immediate finding of note is that the fingerprint, MCS fusion and hyperstructure results were markedly complementary. MCS fusion and hyperstructures became less complementary as the value of β neared 0.9 or 0.95 (the ones that retrieved higher BEDROC scores), though were still somewhat complementary with roughly only half of the compounds being in common in the matched lists. The different rankings between fingerprints and graph-matching has been shown previously by Raymond and Willett (2002a), who showed that, on average, 30% of the active compounds retrieved from graph similarity and fingerprint similarity methods were different. Perhaps unsurprisingly, the overlaps between more different values of β also decreased, though this may have been due to a difference in recall rate, rather than actual complementarity. These results were ultimately the reason behind the fusion of FP MAX, MCS MAX and HS 90 in various forms as discussed later in this section. However, it is unknown how other values of β could affect performance,



(c)
 Figure 4.11. Compounds involved in the MDDR COX inhibitors search. 4.11a shows the molecules used to construct the hyperstructure (numbered arbitrarily). 4.11b shows the top five-ranked active compounds retrieved by the hyperstructure, with their ranks displayed. 4.11c shows the top five-ranked active compounds retrieved by FP MAX, with their ranks displayed.

though one would expect that lowering the value would improve compound diversity, at the cost of compound and scaffold recall.

4.3.5 Time and Compression Results

Table 4.6 shows the performance times of the hyperstructure and MCS searches, and Table 4.7 shows the extent of the compressions of the input molecules. Comparing with the hyperstructures, the typical search time requirement in MDDR was roughly two orders of magnitude greater than fingerprint searches, though for WOMBAT the time requirements were greater than this, and the opposite can be observed for MUV. The hyperstructure searches were consistently faster than MCS in all datasets

targetID	Construction		Database Searching			
	HS	HS	MCS	FP	HS:MCS	HS:FP
6233	1.467	906	1,847	15.2	0.491	59.5
6235	1.033	1,408	2,264	59.2	0.622	23.8
6245	0.633	1,089	2,019	14.5	0.540	75.3
7701	0.901	1,484	2,120	14.6	0.700	101.4
31420	1.585	2,433	4,940	33.6	0.493	72.4
31432	1.627	2,134	3,392	19.5	0.629	109.6
37110	1.036	1,858	2,941	17.2	0.632	107.8
42731	2.312	3,169	3,975	44.1	0.797	71.8
71523	2.258	2,952	4,043	23.5	0.730	125.5
78331	1.077	1,340	2,189	20.4	0.612	65.7
78374	1.013	1,993	2,575	20.9	0.774	95.4
mean	1.358	1,888	2,937	25.7	0.638	82.6

(a) *MDDR*

targetID	Construction		Database Searching			
	HS	HS	MCS	FP	HS:MCS	HS:FP
1	1.632	9,860	49,260	20.5	0.200	480.1
2	0.566	5,973	32,919	20.3	0.181	294.7
3	3.606	56,539	141,896	25.6	0.398	2,208.6
4	1.437	4,725	19,572	19.9	0.241	237.5
5	1.414	6,523	23,918	16.3	0.273	401.4
6	0.379	1,997	14,952	17.1	0.134	116.7
7	0.910	4,150	27,770	16.5	0.149	252.1
8	0.479	3,985	19,724	16.5	0.202	241.5
9	1.198	9,849	48,361	16.2	0.204	608.9
10	1.647	18,469	87,894	18.4	0.210	1,002.8
11	3.321	12,575	81,207	23.0	0.155	547.0
12	0.485	13,756	38,933	22.3	0.353	617.0
13	3.579	4,775	36,093	19.8	0.132	241.3
14	2.412	3,639	34,018	28.7	0.107	126.8
mean	1.648	11,201	46,894	20.1	0.210	526.9

(b) *WOMBAT*

targetID	Construction		Database Searching			
	HS	HS	MCS	FP	HS:MCS	HS:FP
466	0.669	190	1,470	6.7	0.129	28.3
548	0.516	125	1,204	6.5	0.103	19.2
600	0.602	160	1,410	7.5	0.114	21.2
644	0.507	130	1,177	7.0	0.110	18.5
652	0.609	178	1,347	6.6	0.132	26.8
689	0.566	191	1,469	7.8	0.130	24.5
692	0.562	199	1,187	6.6	0.168	30.3
712	0.394	204	1,406	6.7	0.145	30.5
713	0.512	191	1,396	6.6	0.137	29.0
733	0.574	204	1,515	6.8	0.135	29.8
737	0.643	222	1,841	10.3	0.121	21.6
810	1.369	225	1,387	7.3	0.162	30.9
832	0.544	208	1,792	6.7	0.116	31.1
846	0.402	159	1,570	6.3	0.101	25.2
852	0.376	259	1,769	8.0	0.146	32.4
858	0.605	237	2,178	6.9	0.109	34.3
859	0.528	269	2,426	8.0	0.111	33.9
mean	0.587	197	1,561	7.2	0.128	27.5

(c) *MUV*

Table 4.6

Time performances on the constructed hyperstructures, as well as for MCS fusion and Fingerprint searching for each class in the MDDR dataset, reported in seconds. Fingerprint similarity here includes the time taken to create the fingerprints. HS:MCS is the fraction of time that the hyperstructure search took compared to the MCS search, and likewise for HS:FP for hyperstructures against fingerprints.

targetID	Compression		Size	
	C_B	C_A	B_H	A_H
6233	0.376	0.533	63	40
6235	0.338	0.498	76	46
6245	0.333	0.485	75	47
7701	0.310	0.405	84	60
31420	0.388	0.610	115	69
31432	0.360	0.580	99	55
37110	0.337	0.516	100	61
42731	0.326	0.501	123	73
71523	0.297	0.420	124	81
78331	0.351	0.534	68	41
78374	0.309	0.460	89	55
mean	0.339	0.504	92	57

(a) *MDDR*

targetID	Compression		Size	
	C_B	C_A	B_H	A_H
1	0.391	0.537	130	90
2	0.347	0.482	74	49
3	0.291	0.386	178	126
4	0.273	0.340	153	115
5	0.320	0.478	136	83
6	0.352	0.514	71	44
7	0.320	0.402	113	82
8	0.307	0.431	99	64
9	0.283	0.346	144	109
10	0.368	0.624	94	51
11	0.223	0.260	163	131
12	0.349	0.495	92	61
13	0.303	0.421	87	58
14	0.359	0.516	78	49
mean	0.321	0.445	115	79

(b) *WOMBAT*

targetID	Compression		Size	
	C_B	C_A	B_H	A_H
466	0.369	0.602	80	44
548	0.377	0.640	66	35
600	0.357	0.559	77	44
644	0.356	0.533	64	39
652	0.347	0.548	80	46
689	0.372	0.607	74	41
692	0.364	0.589	66	37
712	0.358	0.538	74	45
713	0.323	0.494	81	49
733	0.353	0.522	73	45
737	0.372	0.624	76	41
810	0.326	0.476	68	42
832	0.338	0.539	77	44
846	0.375	0.597	59	34
852	0.337	0.471	75	49
858	0.356	0.549	73	43
859	0.364	0.586	74	42
mean	0.356	0.557	73	42

(c) *MUV*

Table 4.7

Compression statistics on the constructed hyperstructures. C_B is the bond compression; C_A is for atom compression; B_H represents the number of bonds in the hyperstructure and A_H , the number of atoms.

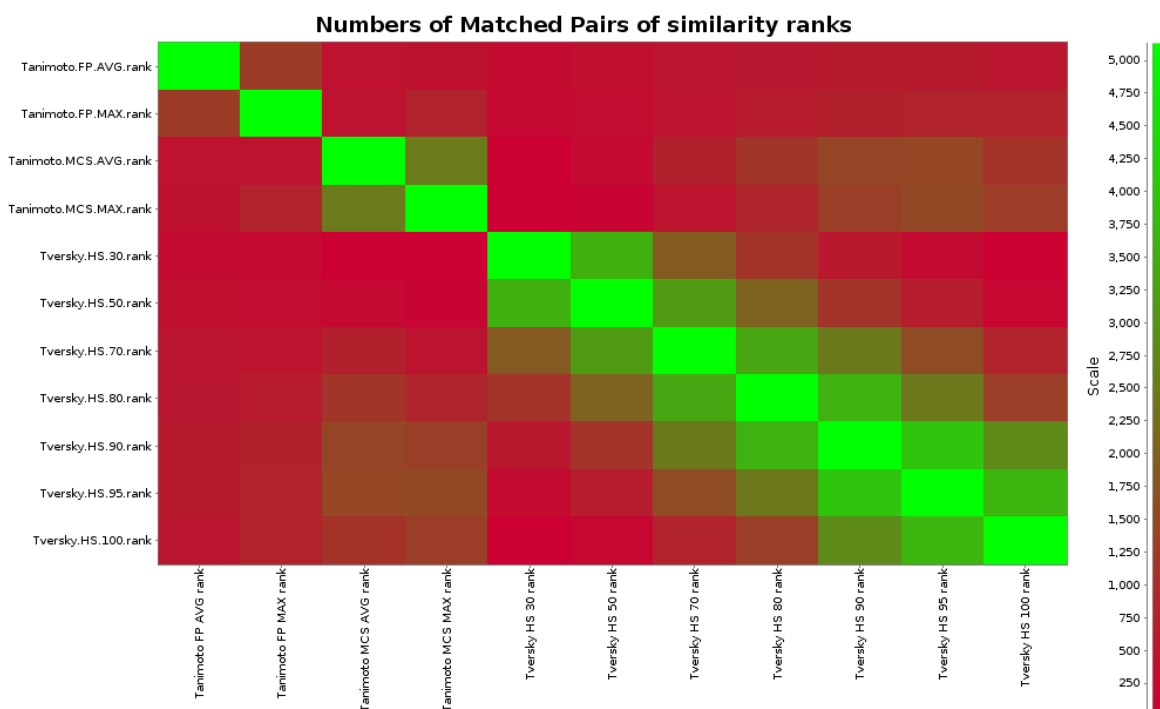


Figure 4.12. Heat map representing the numbers of matched pairs for the ranks of the MDDR compounds tested with the methods shown, in this case for the Cyclooxygenase Inhibitors class. Red corresponds to low overlap, whereas green corresponds to high overlap.

- The speed-up for hyperstructure searches in MUV and WOMBAT being between 0.1 and 0.4, depending on the activity class. The time difference between MCS and hyperstructure searching was less apparent in MDDR, though hyperstructures were still consistently faster. Unsurprisingly there is a positive relationship between the number of bonds and the search time, with a similar relationship existing for construction time. The size of the hyperstructure does not always increase with time performance. An example is the 163 bond hyperstructure for WOMBAT's target ID of 11, whose search ran faster than the 94 bond hyperstructure of target ID 10. One should note, however, that this reflects more on the inexact nature of the MCS algorithm used, as opposed to the nature of the hyperstructure. One would expect a conventional exact MCS algorithm to have exponential time scaling, with graph size. From internal correlation calculations (not shown), we found that there was no significant relationship between bond compression and virtual screening performance.

4.3.6 Similarity Fusion

When looking at BEDROC scores, the Wilcoxon tests (Figure 4.6 and Table E.1) and the mean ranks (Figures 4.8, 4.9 and 4.10, and Table F.1), none of the fusion methods outperformed FP MAX, though one should note that the rank difference is insignificant when comparing FPMAX with FPMAX MC-

method	Mean(\bar{x}_{F1})	sd(\bar{x}_{F1})	Mean(BEDROC)	sd(BEDROC)
FP AVG	2.766	0.249	0.932	0.099
FP MAX	2.553	0.269	0.860	0.069
FPMAX HS90	2.712	0.302	0.817	0.065
FPMAX MCSMAX	2.455	0.238	0.839	0.034
FPMAX MCSMAX HS90	2.653	0.324	0.831	0.041
MCS AVG	2.904	0.414	0.762	0.053
MCS MAX	2.202	0.236	0.601	0.061
MCSMAX HS90	2.454	0.357	0.668	0.078
Tversky HS 100	1.151	0.702	0.084	0.108
Tversky HS 30	1.436	0.152	0.348	0.170
Tversky HS 50	1.648	0.222	0.487	0.211
Tversky HS 70	2.005	0.273	0.624	0.179
Tversky HS 80	2.132	0.268	0.662	0.168
Tversky HS 90	2.258	0.443	0.512	0.138
Tversky HS 95	2.377	0.444	0.368	0.134

(a) Activity class 31420 (Renin inhibitors)

method	Mean(\bar{x}_{F1})	sd(\bar{x}_{F1})	Mean(BEDROC)	sd(BEDROC)
FP AVG	2.254	0.500	0.289	0.041
FP MAX	2.855	0.419	0.526	0.057
FPMAX HS90	2.507	0.873	0.417	0.082
FPMAX MCSMAX	3.135	0.641	0.576	0.055
FPMAX MCSMAX HS90	2.618	0.653	0.490	0.074
MCS AVG	2.408	0.754	0.156	0.028
MCS MAX	3.406	0.921	0.474	0.071
MCSMAX HS90	2.299	0.758	0.351	0.092
Tversky HS 100	2.025	0.546	0.223	0.033
Tversky HS 30	1.583	0.804	0.043	0.036
Tversky HS 50	1.653	0.834	0.074	0.052
Tversky HS 70	1.965	0.925	0.121	0.068
Tversky HS 80	2.021	0.783	0.157	0.078
Tversky HS 90	1.977	0.550	0.233	0.090
Tversky HS 95	1.946	0.414	0.234	0.055

(b) Activity class 78331 (Cyclooxygenase inhibitors)

Table 4.8

Virtual screening statics for replicates of two MDDR activity classes. "sd" in this case represents the standard deviation.

SMAX (Table E.1). The fusion of all three methods, FPMAX MCSMAX HS90, was not significantly worse than fingerprints alone, though the grossly increased time requirement and lack of benefit in diversity retrieval makes the method redundant. The same observations can be made when concerning $EF_{FF1\%}$ (Figure 4.7 and Table E.2). By contrast, the best of these fusions for \bar{x}_{F1} were FPMAX HS90

and MCSMAX HS90, noting that in all datasets the results were significant, in contrast to $EF_{FF1\%}$. The summary plots (Figures 4.8, 4.9 and 4.10) also highlight the trade-off between BEDROC and molecules retrieved per framework. If one is interested only in retrieving active compounds with no regard for diversity, then FP MAX represents the best choice overall. However, if one is more interested in retrieving a different set of scaffolds (and thus reducing analogue retrieval), one of the hyperstructure fusion techniques, notably FP MAX HS90 (due to being faster than MCSMAX HS90), would be more suitable, though at a cost in recall.

4.3.7 Variability

To assess the variability of the methods tested, the most diverse (Cyclooxygenase inhibitors, class 78331) and least diverse (Renin inhibitors, class 31420) activity classes were chosen from the MDDR database. Five similarity searches were performed for each activity class, each time using a different random seed for the diversity selection of the reference molecules. Due to time constraints it was unfeasible to obtain a greater number of replicates, so statistical tests cannot reliably be performed on these results. The results for virtual screening variability are shown in Table 4.8. For renin inhibitors, Hyperstructures alone displayed a greater variability in BEDROC than any of the other methods. Interestingly the fusion methods all have more consistent BEDROC scores, comparable to those of fingerprints - implying that they were less dependent on the reference compounds used. A similar pattern is also observable for the cyclooxygenase inhibitors, where (relative to the mean) the variability for hyperstructures was high. Variability in the framework statistic $\bar{x}_{F1\%}$ amongst hyperstructures was also higher than other methods, the other methods generally having more similar variabilities, for both activity classes. These results at least suggest that hyperstructures are better utilised in fusion with fingerprints to make them less dependent on the reference compounds used, regardless of whether the dataset is homogeneous or heterogeneous.

4.3.8 Physicochemical Properties of Retrieved Compounds

The physicochemical properties shown in Table 4.9 (MDDR only) highlight some statistically significant differences in the molecules between hyperstructure, MCS and fingerprint searches. Of immediate note is that hyperstructures retrieve significantly larger (more atoms) active molecules than fingerprints. The size difference is apparent, in spite of the fact that the Tversky coefficient has been used in order to attempt to reduce size bias. However, the size difference is only about 1.5 atoms, despite statistical significance. In addition to being larger, the molecules are more complex

Method	Complexity	Largest Chain	Rotatable Bonds	Heavy Atoms	Heteroatoms	Rings
FP MAX	3570	15.022	11.511	32.253	7.714	3.874
MCS MAX	3146	13.188	10.554	29.995	6.793	3.655
Tversky HS 90	3969	16.469	12.486	33.913	7.632	3.873
p_MCS MAX	0.002	0.005	0.054	0.005	0.005	0.175
p_Tversky HS 90	0.010	0.042	0.083	0.032	1.000	0.765

Table 4.9

Mean values of physicochemical properties of the actives of the top 1% ranked lists for given methods, derived from the MDDR dataset. Values with the prefix “p_” reflect the p-values from paired 2-tailed Wilcoxon signed rank tests, tested against FP MAX.

and possess larger acyclic chains, but have no significant difference in heteroatom and ring count. This implies that the molecules are less “feature-rich” and more chain-rich. By contrast, the MCS-retrieved active molecules are smaller and less chain-rich, though also possessing significantly less heteroatoms. These observations have been made from the MDDR dataset, and it is thus assumed that the same will apply for the other datasets.

4.4 Conclusions

The results of this investigation showed that both hyperstructures and MCS fusion, at least for the datasets used, are inferior to a conventional fingerprinting method for performing virtual screening (according to BEDROC scores). Both techniques retrieve fewer active compounds, and fewer active scaffolds than fingerprint searching, and also take significantly longer to calculate. It is also noteworthy that hyperstructures are more dependent on the reference compounds used to construct the hyperstructure, than fingerprints and MCS fusion are.

The value of the β parameter does have some influence on both virtual screening recall and the diversity of compounds retrieved. From the Kendall W results, we suggest as a rule of thumb to use a β of 0.9. From this value, it is suggested that substructure-based similarity is preferable for most virtual screening applications. It is unclear why a value of 1.0 performed so badly; evidently some form of size bias is required for accurate similarity. For example, if two compounds of different sizes are perfect substructures of another molecule, then a β of 1.0 will treat both as identical irrespective of their differing sizes. Perhaps in their current states, there are too many inactive molecules that are unintentionally overlapping well with otherwise undesirable elements of the hyperstructure, and thus we have to rely on the difference in numbers of edges in the hyperstructure and database molecules somewhat to improve active/inactive distinction. Related to this hypothesis, N. Brown (2002) claimed

that “rogue ghost substructures” were potentially responsible for the poor relative performance of hyperstructures to fingerprints in his work on similarity searching. Evidently from the diversity results here, the value of β could be adjusted if the experimenter desires a superior diversity, though at the cost in recall.

Hyperstructure searches have a consistent tendency to yield fewer analogues than fingerprint searches, although they have also been shown to have worse scaffold retrieval in general when compared with fingerprints. The complementarity between hyperstructure, MCS fusion and fingerprint ranked lists, suggested that data fusion of the three techniques would be feasible in improving similarity searching. This has somewhat supported the actual results obtained from such fusion, though at a great cost in search time (that is, compared to fingerprints alone). Of note, we have identified that the fusion of fingerprints and hyperstructures (at a β of 0.9 in similarity searching) is the best trade-off of the fusion rules between virtual screening recall, and retrieval of a diverse set of molecular frameworks, though the fusion of fingerprints and MCS yields a recall comparable to fingerprint fusion (though at a lower scaffold retrieval). It is currently unknown as to exactly how a different value of β will affect fusion performance, though it is likely that a lower value will worsen both active compound and active scaffold recall.

There are several potential reasons why hyperstructures perform differently to fingerprints, aside from the worsened ability to find analogues. The most obvious possible cause is the binary nature of graph mapping, which minimises multiple mappings of the same feature compared with fingerprint comparisons (some bits, particularly with radial fingerprints of increasingly large radii, partially represent other bits). This degeneracy could be important, leading to the potential idea of applying edge weighting to hyperstructures based on frequency of overlap (as explored in Chapter 5). It is also unknown what effect ghost substructures have on search results - The effects of ghost substructures are explored in Chapter 5.

Chapter 5

Similarity Searching with Modified Hyperstructures

5.1 Introduction

This work continues with the theme of similarity-based virtual screening from chapter 4. The focus of this chapter is on weighting of hyperstructures and application of topological constraints in similarity searching, again measuring recall and diversity of the methods tested. In addition, the influence of “ghost substructures” mentioned in the chapter 2 will be monitored in this chapter, as it is currently unknown whether they have an effect on similarity searching, or have differing frequencies in active and inactive molecules. One should note that, shortly after completing the experimental work of this chapter, we came across the work of Klinger and Austin (2006). The work in the mentioned article is highly similar to the work described here, but with some interesting differences, which will be elaborated on at the end of this chapter.

5.2 Materials and Methods

The methodology here, unless otherwise stated, is the same as described in chapter 4, including the datasets used, hyperstructure construction methodology, similarity searching and evaluation metrics used.

5.2.1 Monitoring Ghost Substructures

R. D. Brown, Downs, et al. (1994) utilised a fragment dictionary to search for ghosts in hyperstructures. A set of substructures were obtained from the union of all the substructures found (from the dictionary) in the molecules used to build a hyperstructure. These would represent the hyperstructure's substructures. When performing subgraph isomorphism with a query molecule against a hyperstructure, the subgraph isomorphism would only be performed if the hyperstructure contained all of the substructures that the query contained. We believe this has a number of problems for determining ghost substructures:

- The construction of a hyperstructure itself *guarantees* the production of ghost substructures, provided that the hyperstructure is not isomorphic to one of the input molecules. This is because, at the very least, the constructed hyperstructure (if not isomorphic) will be bigger than its input molecules, and thus the largest substructure (i.e. itself) will not feature in the input molecules. It would be better to obtain substructures directly from the hyperstructure, rather than its input molecules, as a greater yield of ghosts should be possible.
- It is highly dependent on the fragment dictionary used (which may not be readily available), and the dictionary itself may not necessarily encapsulate all possible ghosts
- The use of fragment dictionaries require subgraph isomorphism checks for each fragment. When using thousands of fragments this can be a time consuming process, particularly in large database searching (despite recent improvements in time complexity for subgraph isomorphism tests).

When performing similarity searching, we are only interested in ghost substructures in the hyperstructure that a query molecule maps to. The MCS provides a basis for finding ghosts in the hyperstructure which are relevant to the query. We have adopted a different technique for finding ghost substructures, based on the methodology of extended connectivity fingerprints. Figure 5.1 shows an example of how ghost substructures are enumerated for a hyperstructure. For each atom in a molecule, all bonds up to a fixed path length from the atom are included in a radial fashion to form a subgraph (i.e. a breadth-first search). Duplicate subgraphs are then removed on the basis of the identifiers of the bonds in the MCS. This means that isomorphic substructures are allowed, provided that they originate from different parts of the MCS. Given that each bond in the hyperstructure has originated

from at least one (but often more) bond from the input molecules, we can perform an intersection of the molecular origins of each bond in a given substructure. If the intersection yields an empty set, then the substructure is a ghost (refer to chapter 2 for more examples on ghost substructures). An example of the same technique being applied to the MCS of a hyperstructure and query is shown in Figure 5.2, where the bond origin information from the hyperstructure is copied to the MCS. In this work, the radius for the technique is set to 4 for finding radial ghost substructures.

5.2.2 Bond Weighting

This concept was applied by N. Brown et al. (2003) in the concept of activity-weighted hyperstructures (or hypermolecules), though in the concept of substructural analysis (see chapter 2 for further details). In this chapter instead, it is applied in similarity searching. As with the cited work, bonds are weighted based on their frequency of overlap, though in this chapter, only active molecules are used during construction. The weight of a bond in the weighted hyperstructure is also proportional to the number of MCSs it was in. An example is presented in Figure 5.3.

Five weighting schemes (W1-W5) used by Arif, Holliday, and Willett (2009) and Holliday et al. (2012) have been used to generate “similarity weights” for hyperstructure bonds. A set of weights w in the hyperstructure, with a weight w_j for each hyperstructure bond j , was assigned in the construction process (i.e. the bond’s overlap frequency). Modification through one of the weighting schemes yielded the final weight y_j . The weighting schemes are as follows, starting with the first weight

$$W1 : y_j = 1$$

which is a binary weight, which otherwise ignores any weights assigned to the bond during hyperstructure construction. In other words, all bonds have equal weights, and otherwise mirror the results obtained in chapter 4. W2 by contrast is just the original assigned weight of the bond:

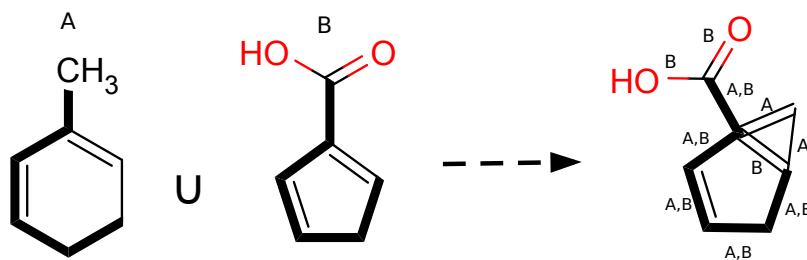
$$W2 : y_j = w_j$$

The next two weighting schemes are simple modifications, being the natural logarithm and the square root respectively

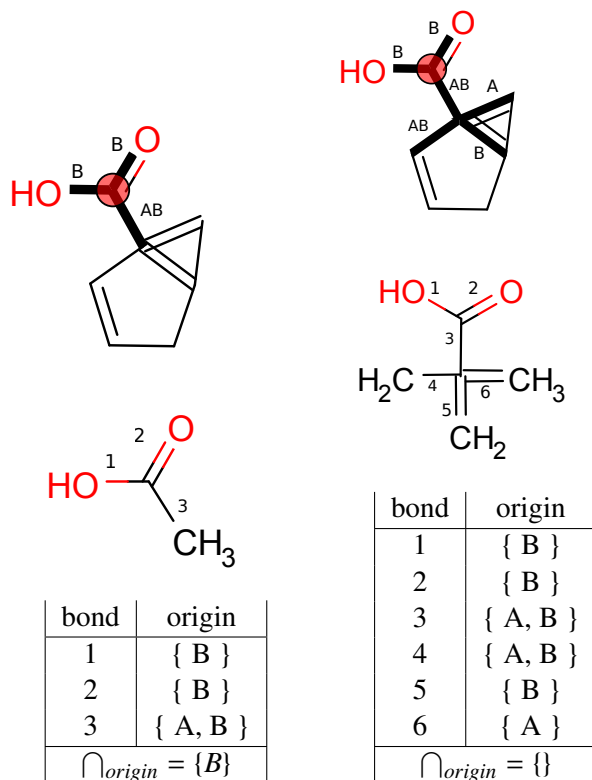
$$W3 : y_j = \ln(w_j)$$

$$W4 : y_j = \sqrt{w_j}$$

The logarithmic term in W3 exists mainly to remove features weighted at 1 (log of 1 gives 0), yet the magnitude between higher weights matters less with this scheme. W4 simply lessens the effect of



(a) Hyperstructure construction between molecules A and B. Bold edges indicate the MCS of A and B. Bond weights on the hyperstructure indicate which molecule the bond originated from (AB means both A and B).



(b) substructure generation
at a radius of 1

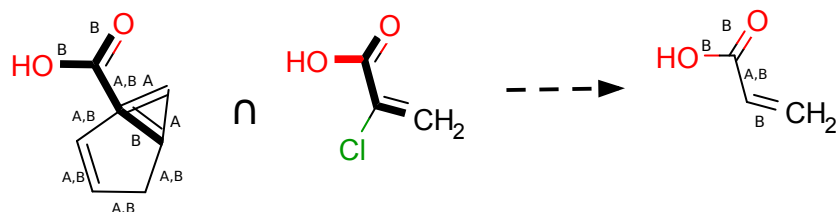
(c) substructure generation
at a radius of 2

Figure 5.1. Finding ghost substructures in a hyperstructure. In parts 5.1b and 5.1c, bold edges in the hyperstructure represent the identified substructure from the central atom (circled). A copy of the subgraph is shown with the atoms labelled, with an accompanying table to show the set intersection process.

more highly weighted bonds. W5 is a normalisation of W2 based on the maximum weight found in the hyperstructure:

$$W5 : y_j = 0.5 + 0.5 \frac{w_j}{\max\{w\}}$$

This weighting scheme has been successful in text-based information retrieval (Salton & Buckley, 1988).



(a) MCS between the hyperstructure in Figure 5.1 with a database molecule. The bond weights have been carried over from the hyperstructure to the MCS.

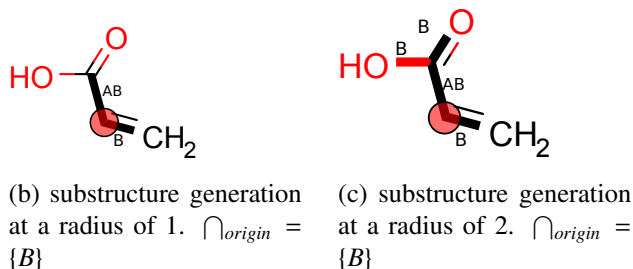


Figure 5.2. Finding ghost substructures in the MCS (as opposed to the hyperstructure) between a hyperstructure and a database molecule. In this particular example, no ghost substructures exist as all intersections are non-empty.

The gain in active compound recall has not generally been shown to have a great benefit over binary weighting, when using weighting schemes with extended connectivity fingerprints for virtual screening. Despite this, W4 and W5 have consistently been shown to outperform W1 (in cases where weights are applied to both reference and database molecules), whereas W3 is ranked somewhat lower than W1, with W2 consistently ranking worst. By contrast, when only the reference structures are weighted, W2 and W4 have generally outperformed W1, whereas W3 and W5 have usually been inferior to W1 (Arif, Holliday, & Willett, 2009; Holliday et al., 2012). In this study, the database molecule bonds all had a weight of 1 (in other words, W1 is used on the bonds), so one would expect that W2 and W4 would outperform W1 with hyperstructures. This, of course, assumes that fragment-based virtual screening behaves in a similar manner to hyperstructure similarity searching, which chapter 4 has otherwise shown to not be the case.

For similarity searching, a continuous form of the Tversky coefficient has been used:

$$S_{Tv} = \frac{\sum_{k \in MCS} x_k y_k}{\beta \left(\sum_{i \notin MCS} x_i^2 \right) + (1 - \beta) \left(\sum_{j \notin MCS} y_j^2 \right) + \left(\sum_{k \in MCS} x_k y_k \right)}$$

where x_i represents the weight of bond i in the database molecule and y_j the weight in the reference molecule (or hyperstructure) for bond j . The numerator of the equation is the weighted sum of the

bonds in both molecules, which feature in the MCS. By contrast, $\left(\sum_{i \notin MCS} x_i^2\right)$ and $\left(\sum_{j \notin MCS} y_j^2\right)$ represent the squared weights for the bonds in the relevant molecules that are not in the MCS. As $x_i = 1$ in this study, the equation can be simplified to

$$S_{Tv} = \frac{\sum_{k \in MCS} y_k}{\beta d + (1 - \beta) \left(\sum_{j \notin MCS} y_j^2 \right) + \left(\sum_{k \in MCS} y_k \right)}$$

where d is the number of bonds in the database molecule that are not present in the MCS.

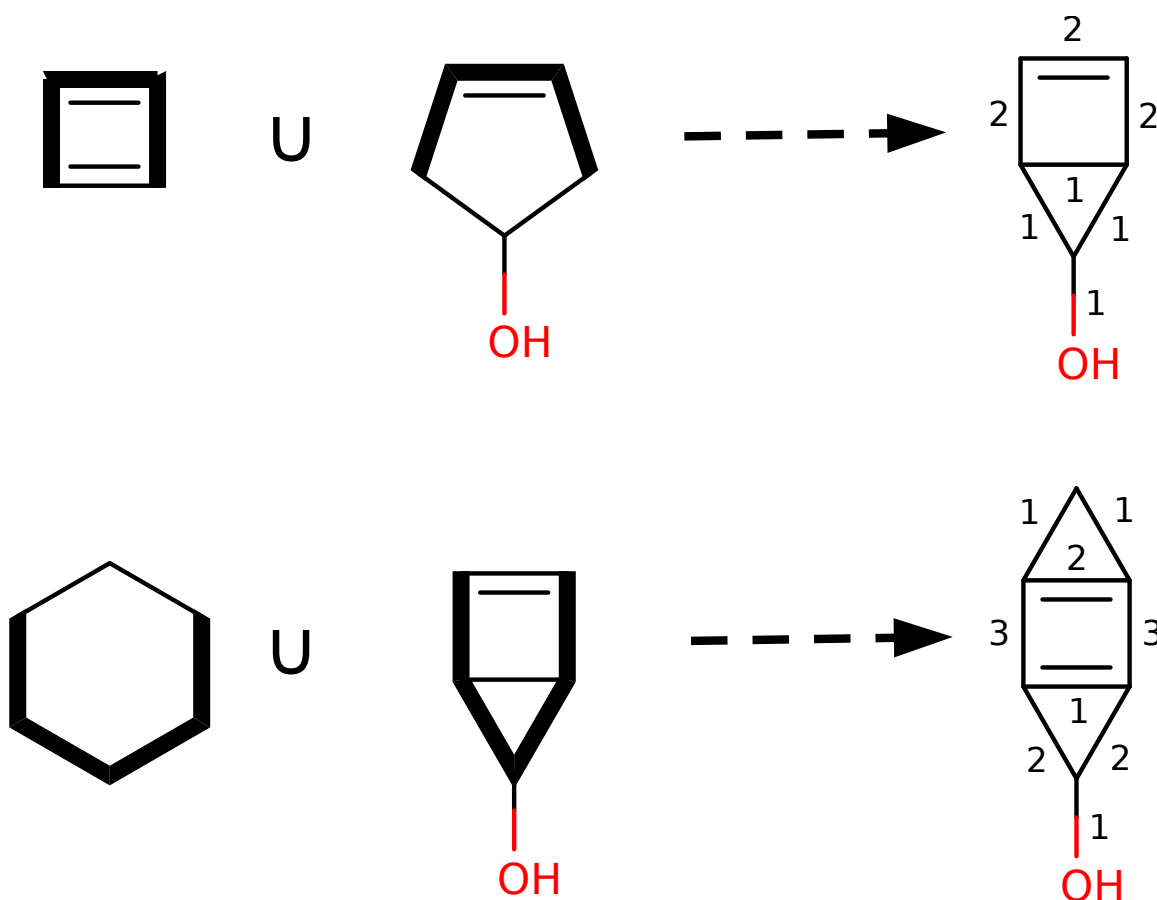


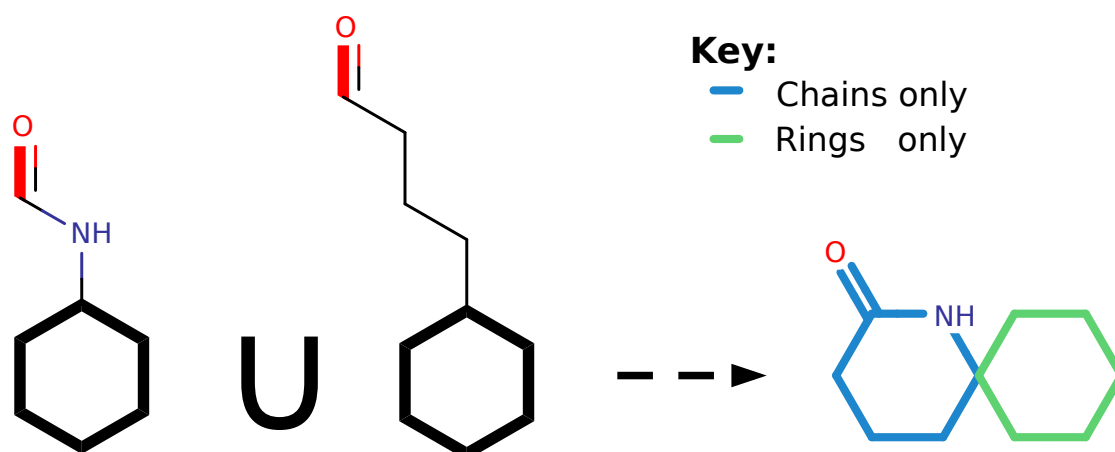
Figure 5.3. Example of hyperstructure construction with bond weighting, based on the frequency of overlap. Bold edges represent the MCS between two molecules, and numbers on the hyperstructures (right-most graphs) represent bond frequency weights. The order of molecules in this example has been arbitrarily chosen.

Hyperstructure search methods will be named as HS_90_ a , where a is a number from 1 to 5 that identifies the weight scheme used, and 90 represents a β value of 0.9. FP denotes fingerprint group fusion as used from chapter 4, and FP HS90_ a is SUM rank fusion of FP with HS_90_ a . As a value of 0.9 was found to be most effective for virtual screening in chapter 4, β was set to 0.9 for this

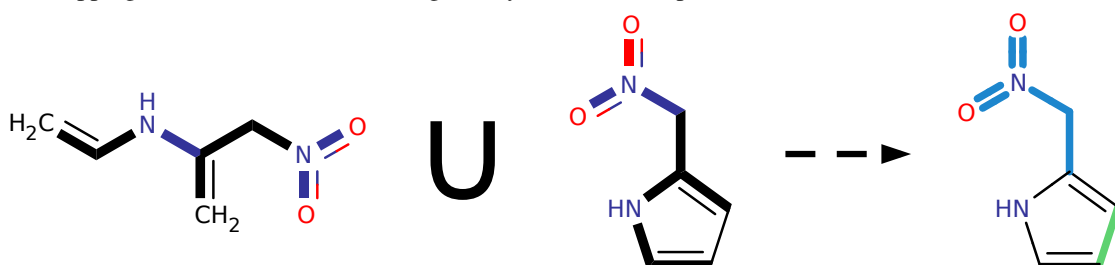
study.

5.2.3 Topology Type Matching

In an attempt to reduce the number of ghost substructures present between the MCS of the hyperstructure and query molecule, we decided to implement a feature that remembers and enforces the topology type of the bond in the hyperstructure. We hypothesised that this will have an effect on the number of ghost substructures mapped, for the creation of new rings during the hyperstructure process is one common way of producing ghost substructures. However, we also expect it to have an impact on the scaffold hopping ability of the search process, being that ghost substructures should theoretically be responsible for the ability to retrieve new scaffolds. Namely, whether the bond originates entirely from chains, rings, or both, in the input molecules. Examples are presented in Figure 5.4. It should be noted that the construction order (that is, the order of input molecules used) is unchanged - topology information is ignored during the sorting process for input molecules.



(a) In this example, a ring in the hyperstructure (far right molecule) has been constructed from two chains overlapping. Note however that this ring is only allowed to map to chain molecules.



(b) An example with bonds that map both chains and rings (black bonds)

Figure 5.4. Topology type enforcement examples. MCS bonds are shown in bold for the input graphs.

5.3 Results and Discussion

5.3.1 Virtual Screening Recall Performance

Method	Mean Rank	Wilcoxon p	
		vs w1	vs FP MAX
HS 90 w1	4.545	1.00	$9.77 \cdot 10^{-4}$
HS 90 w2	1.818	$1.86 \cdot 10^{-2}$	$9.77 \cdot 10^{-4}$
HS 90 w3	3.364	$5.37 \cdot 10^{-2}$	$9.77 \cdot 10^{-4}$
HS 90 w4	3.364	0.28	$9.77 \cdot 10^{-4}$
HS 90 w5	3	$2.07 \cdot 10^{-2}$	$9.77 \cdot 10^{-4}$
FP HS90 w1	8.909	1.00	$8.30 \cdot 10^{-2}$
FP HS90 w2	6.182	$1.37 \cdot 10^{-2}$	$4.88 \cdot 10^{-3}$
FP HS90 w3	8.182	0.18	$4.88 \cdot 10^{-3}$
FP HS90 w4	8.364	0.21	$2.44 \cdot 10^{-2}$
FP HS90 w5	8.455	0.12	$1.37 \cdot 10^{-2}$
FP	9.818	1.00	1.00

(a) *BEDROC*. Kendall's W (tested on the ranks of each activity class) = 0.659, $p = 1.28e-39$

Method	Mean Rank	Wilcoxon p	
		vs w1	vs FP MAX
HS 90 w1	3.818	1.00	$1.95 \cdot 10^{-3}$
HS 90 w2	2.818	0.58	$9.77 \cdot 10^{-4}$
HS 90 w3	3.091	0.37	$9.77 \cdot 10^{-4}$
HS 90 w4	4	0.52	$9.77 \cdot 10^{-4}$
HS 90 w5	2.727	0.12	$9.77 \cdot 10^{-4}$
FP HS90 w1	8.091	1.00	$6.74 \cdot 10^{-2}$
FP HS90 w2	7.455	0.97	0.12
FP HS90 w3	8.727	0.64	$2.44 \cdot 10^{-2}$
FP HS90 w4	9.091	0.41	0.64
FP HS90 w5	6.636	$5.37 \cdot 10^{-2}$	$4.88 \cdot 10^{-3}$
FP	9.545	1.00	1.00

(b) *EF_{FF1}*. Kendall's $W = 0.600$, $p = 9.47e-35$

Table 5.1

Virtual screening results for the MDDR dataset. The "Mean Rank" column shows the mean of the ranks, of all the methods. Cells in bold typeface indicate the weighting scheme that performed best for a given similarity method, and cells shaded in grey indicate the maximum rank across all methods. The "Wilcoxon p " columns presents the p -values for two-tailed Wilcoxon signed-rank tests, "vs w1" for the method being tested against the w1 equivalent, and "vs FP MAX" for the given method against FP.

Method	Mean Rank	Wilcoxon p	
		vs w1	vs FP MAX
HS 90 w1	1.941	1.00	$1.53 \cdot 10^{-5}$
HS 90 w2	3.706	$3.51 \cdot 10^{-2}$	$7.63 \cdot 10^{-5}$
HS 90 w3	4.794	$3.81 \cdot 10^{-4}$	$5.48 \cdot 10^{-4}$
HS 90 w4	4.529	$2.58 \cdot 10^{-3}$	$6.52 \cdot 10^{-4}$
HS 90 w5	4.941	$1.53 \cdot 10^{-5}$	$3.05 \cdot 10^{-5}$
FP HS90 w1	4.471	1.00	$1.53 \cdot 10^{-5}$
FP HS90 w2	6.882	0.17	$1.53 \cdot 10^{-4}$
FP HS90 w3	8.294	$8.39 \cdot 10^{-4}$	$2.14 \cdot 10^{-4}$
FP HS90 w4	7.794	$1.06 \cdot 10^{-2}$	$2.90 \cdot 10^{-4}$
FP HS90 w5	8.529	$1.53 \cdot 10^{-5}$	$1.51 \cdot 10^{-3}$
FP	10.118	1.00	1.00

(a) *BEDROC*. Kendall's $W = 0.243$, $p = 3.35e-17$

Method	Mean Rank	Wilcoxon p	
		vs w1	vs FP MAX
HS 90 w1	4.853	1.00	$1.10 \cdot 10^{-3}$
HS 90 w2	4	0.15	$8.51 \cdot 10^{-4}$
HS 90 w3	4.588	0.67	$1.33 \cdot 10^{-3}$
HS 90 w4	4.676	0.53	$1.09 \cdot 10^{-3}$
HS 90 w5	4.059	0.10	$7.25 \cdot 10^{-4}$
FP HS90 w1	7.618	1.00	$1.07 \cdot 10^{-2}$
FP HS90 w2	6.059	0.18	$5.64 \cdot 10^{-3}$
FP HS90 w3	7.088	0.47	$1.31 \cdot 10^{-2}$
FP HS90 w4	6.647	0.33	$4.09 \cdot 10^{-3}$
FP HS90 w5	7.147	0.93	$5.08 \cdot 10^{-3}$
FP	9.265	1.00	1.00

(b) *EF_{FF1}*. Kendall's $W = 0.157$, $p = 5.71e-08$

Table 5.2

Virtual screening results results for the MUV dataset.

Tables 5.1a, 5.2a and 5.3a display *BEDROC* mean ranks and statistical significance for each weight for the activity classes of the respective databases. Likewise, Tables 5.1b, 5.2b and 5.3b display the same statistics derived from *EF_{FF1}*. The raw virtual screening statistics are in Appendix G. It

Method	Mean Rank	Wilcoxon p		Method	Mean Rank	Wilcoxon p	
		vs w1	vs FP MAX			vs w1	vs FP MAX
HS 90 w1	1.857	1.00	$1.22 \cdot 10^{-4}$	HS 90 w1	4.214	1.00	$2.44 \cdot 10^{-4}$
HS 90 w2	1.5	$7.85 \cdot 10^{-2}$	$1.22 \cdot 10^{-4}$	HS 90 w2	1.464	$2.66 \cdot 10^{-3}$	$1.22 \cdot 10^{-4}$
HS 90 w3	4.821	$1.22 \cdot 10^{-4}$	$1.22 \cdot 10^{-4}$	HS 90 w3	4.821	0.43	$8.54 \cdot 10^{-4}$
HS 90 w4	3.429	$3.05 \cdot 10^{-3}$	$1.22 \cdot 10^{-4}$	HS 90 w4	2.929	$3.76 \cdot 10^{-2}$	$1.22 \cdot 10^{-4}$
HS 90 w5	4.607	$1.22 \cdot 10^{-4}$	$1.22 \cdot 10^{-4}$	HS 90 w5	3.857	0.30	$8.54 \cdot 10^{-4}$
FP HS90 w1	6	1.00	$1.22 \cdot 10^{-4}$	FP HS90 w1	7.893	1.00	$1.07 \cdot 10^{-2}$
FP HS90 w2	6.214	0.78	$1.22 \cdot 10^{-4}$	FP HS90 w2	5.571	$2.32 \cdot 10^{-3}$	$1.22 \cdot 10^{-4}$
FP HS90 w3	9.536	$1.22 \cdot 10^{-4}$	$5.80 \cdot 10^{-2}$	FP HS90 w3	9.143	0.14	0.14
FP HS90 w4	8.321	$3.66 \cdot 10^{-4}$	$2.44 \cdot 10^{-4}$	FP HS90 w4	7.964	0.90	$5.25 \cdot 10^{-3}$
FP HS90 w5	9.143	$1.22 \cdot 10^{-4}$	$2.02 \cdot 10^{-2}$	FP HS90 w5	8.036	0.63	$5.04 \cdot 10^{-2}$
FP	10.571	1.00	1.00	FP	10.107	1.00	1.00

(a) *BEDROC*. Kendall's $W = 0.763$, $p = 2.54e-66$ (b) *EF_{FF1}*. Kendall's $W = 0.724$, $p = 6.15e-62$

Table 5.3

Virtual screening results for the WOMBAT dataset.

is clear from the grey-shaded rows in each table that overall, FP again outperformed all other techniques used for the datasets, and significantly so judging from the p -values. The vast majority of the other techniques in the tables were significantly worse than FP, and there was no clear pattern amongst the small number of exceptions which were not significantly different than FP. So far, this remains consistent with the findings of Chapter 4, where hyperstructure similarity was generally outperformed by fingerprint similarity. Despite this negative outcome, some interesting patterns can be observed in MDDR and WOMBAT. The MUV results by contrast seem to show no apparent benefit in any weighting scheme, and there appears to be little performance differences between hyperstructure search techniques (excluding the fusion technique).

Upon analysis of the *BEDROC* scores of MDDR and WOMBAT, we note from the significant agreement in ranks from Kendall's W , that the W3 scheme was overall ranked better than W1 in recall for all the hyperstructure searches. In MUV and WOMBAT, the paired Wilcoxon tests show that W3 was significantly better than W1, whereas MDDR yielded a result of borderline significance at the 0.05 threshold. It was also observed that W4 was competitive with W1, whilst the W2 scheme ranked worst of all techniques. Similar conclusions can be drawn for scaffold retrieval. Internal tests were performed on other values of β , but a value of 0.9 was still generally the best to use. Alternative trends were found for different β values, though this is beyond the scope of this study and will not be discussed.

On looking at the results for *EF_{FF1}* (and hence scaffold retrieval), an immediate point to note is that the majority of results for each weighting scheme are not significantly different from W1. This is in

contrast to BEDROC, where clear advantages (and disadvantages) to W1 can be observed. W3 only ranked as better than W1 in WOMBAT (and even then, not significantly so). In the other datasets, W1 remained the best option for scaffold retrieval (for hyperstructures alone). The implication from this, is that the choice of weighting scheme evidently had a greater effect on analogue retrieval, as opposed to scaffold retrieval.

The fusion technique FP HS90 was generally ranked above all of the non-fusion hyperstructure methods, for all weighting schemes. For all three datasets, the BEDROC results suggested, however, that no individual weighting scheme performed best for this technique. MDDR highlighted W1 as the best, whereas MUV and MDDR showed that W5 and W3 were significantly better in recall than W1, respectively. In only one instance was this technique not significantly worse than FP - W3, for the WOMBAT dataset for both BEDROC and EF_{FF1} . The same weighting scheme pattern can be observed for scaffold hopping, judging from the EF_{FF1} results, where no weighting scheme significantly outperformed FP.

The results here differ somewhat to the work done on fingerprints (Arif, Holliday, & Willett, 2009; Holliday et al., 2012), where there was a small benefit to using weighting schemes other than W1 when only the reference structure was weighted. W2 performed well with fingerprints and W3 generally didn't, yet in this work no consistent benefit has been observed using weighting schemes, for this particular hyperstructure search technique. The somewhat superior performance of W3 compared to other weighting schemes likely arises from its discounting of bonds with a weight of 1 - theoretically biasing a search towards analogue similarity.

Method	Mean Rank	Wilcoxon p		Method	Mean Rank	Wilcoxon p	
		vs w1	vs FP MAX			vs w1	vs FP MAX
HS 90 w1	4	1.00	$9.77 \cdot 10^{-4}$	HS 90 w1	3.182	1.00	$9.77 \cdot 10^{-4}$
HS 90 w2	1.227	$1.95 \cdot 10^{-3}$	$9.77 \cdot 10^{-4}$	HS 90 w2	2.273	0.24	$9.77 \cdot 10^{-4}$
HS 90 w3	4	0.97	$9.77 \cdot 10^{-4}$	HS 90 w3	4.045	0.15	$9.77 \cdot 10^{-4}$
HS 90 w4	2.909	0.18	$9.77 \cdot 10^{-4}$	HS 90 w4	3.5	0.64	$1.95 \cdot 10^{-3}$
HS 90 w5	3.409	0.10	$9.77 \cdot 10^{-4}$	HS 90 w5	2.455	$4.20 \cdot 10^{-2}$	$9.77 \cdot 10^{-4}$
FP HS90 w1	9	1.00	$4.20 \cdot 10^{-2}$	FP HS90 w1	8.818	1.00	$6.74 \cdot 10^{-2}$
FP HS90 w2	5.727	$1.95 \cdot 10^{-3}$	$9.77 \cdot 10^{-4}$	FP HS90 w2	6.909	0.41	$3.22 \cdot 10^{-2}$
FP HS90 w3	8.818	0.21	$6.84 \cdot 10^{-3}$	FP HS90 w3	9	0.47	0.32
FP HS90 w4	8.091	0.10	$4.88 \cdot 10^{-3}$	FP HS90 w4	8.636	0.97	0.41
FP HS90 w5	8.455	0.32	$6.67 \cdot 10^{-3}$	FP HS90 w5	7.545	$6.84 \cdot 10^{-3}$	$9.77 \cdot 10^{-3}$
FP	10.364	1.00	1.00	FP	9.636	1.00	1.00

(a) BEDROC. Kendall's $W = 0.674$, $p = 7.13e-41$ (b) EF_{FF1} . Kendall's $W = 0.686$, $p = 7.36e-42$

Table 5.4

Virtual screening results for the MDDR dataset when topology type matching is applied.

5.3.1.1 Topology Type Matching. As with the results for non topology type matching, none of the hyperstructure-based techniques outperformed fingerprints (Table 5.4). W3, unlike for the non-topology enforced results (MDDR), was competitive with W1. Although it achieved the same BEDROC ranks as W1, its scaffold recall was superior, albeit not on a significant level. W2, W4 and W5 remain as inferior weighting techniques for both HS 90 and FP HS90. It is unclear whether the topology type enforcement technique retrieved different molecules than the non-enforced technique, seeing that there was little apparent difference in active and scaffold recall between the two techniques. On comparing our approach with that of Klinger and Austin (2006), we note some important differences in methodology:

- Their work specifically sought the maximum-weighted tdMCIS ($\theta = 0$) between a weighted hyperstructure and a query molecule - both in hyperstructure construction and similarity searching. Our approach simply used the weights obtained from the MCES provided by the ChemAxon algorithm, which otherwise finds an unweighted MCES.
- The (symmetric) Wallis coefficient (Wallis et al., 2001) was used, instead of the asymmetric approach in this work.
- The maximum number of compounds used in hyperstructure construction was 5, as opposed to 10 as in this work.

Their approach showed that a weighted hyperstructure actually obtained comparable active compound recalls compared to unweighted ones. Clearly it is beneficial to search for a maximum-weighted MCS (as opposed to an unweighted one), though the topological distance limit clearly has an effect as well. They also noted that the hyperstructure approach was comparable in recall to MAX-rule MCS group fusion, which contradicted our results in Chapter 4. The deletion of edges from the hyperstructure (where the weight is 1) yielded a decrease in active compound recall. The nearest parallel in our work, the application of the W3 weighting scheme, yielded an improvement in recall in two of the three datasets.

5.3.2 Ghost Substructure Information

On analysis of the ghost substructure information in the top 1% ranked compounds of a search method (Table 5.5), we observed that the proportion of ghosts identified was consistently slightly smaller for actives compared to inactives. It is unclear what the significance of this observation is, other than

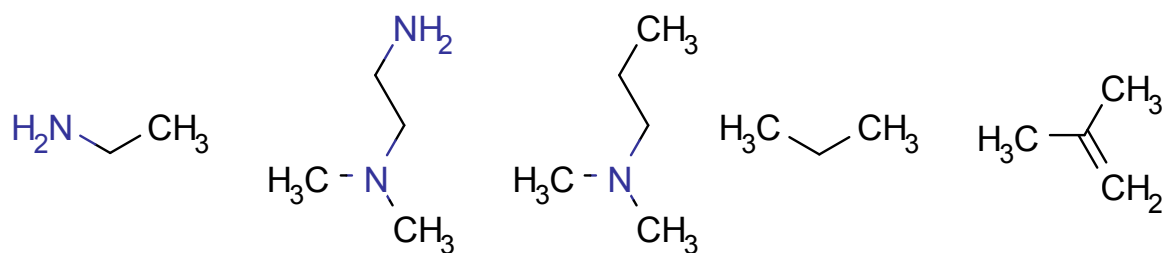
that the highly-ranked inactive molecules generally possess slightly more molecular features that are absent in the molecules used to build the hyperstructures. This may imply that molecules with a greater proportion of ghost substructures should be penalised against in hyperstructure searches, though more investigations are needed to see if this is not simply an artefact of the current search process (for instance, using alternative hyperstructures). This finding would be consistent with the idea of “rogue” ghost substructures being detrimental to virtual screening (N. Brown, 2002).

Method	Gp_A	Gp_I	G_A	G_I	S_A	S_I
HS 90	0.356	0.424	28.353	33.075	79.082	77.791
FP HS90	0.364	0.4	27.912	30.265	75.902	75.433
FP	0.36	0.393	25.871	26.46	71.102	67.671

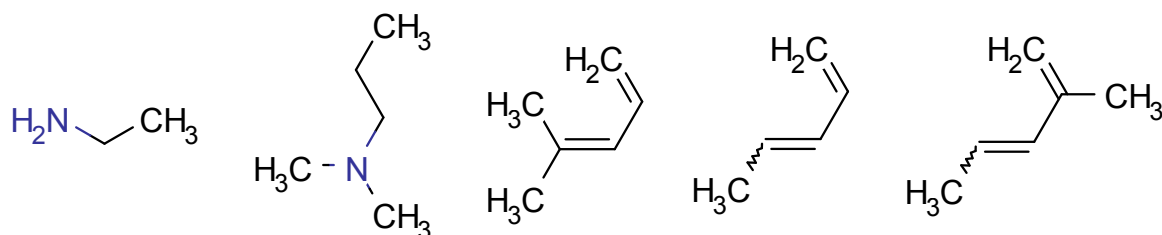
Table 5.5

MDDR Ghost substructure results for compounds in the top 1% ranked compounds for a given search method. Ghost substructures are derived from the MCS between the hyperstructure and database molecule. The subscript A results to active compounds, whereas I identifies inactives (in other words, true positives and false positives respectively, if all compounds in the list were assumed to be active). G is the mean number of unique ghost substructures per compound, S is the mean number of substructures in a compound, and Gp is $\frac{G}{S}$.

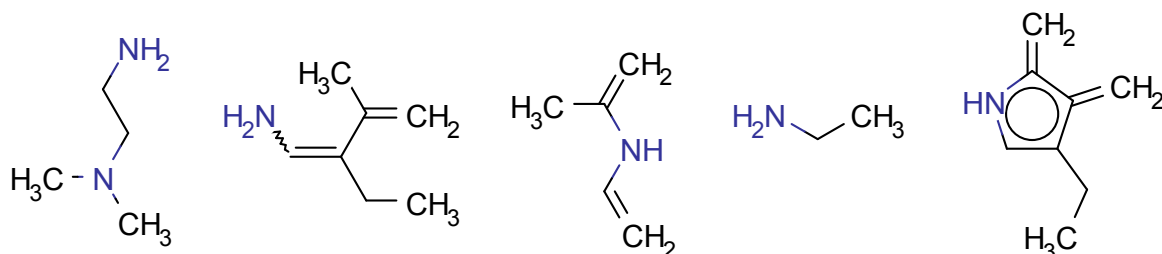
To get a basic idea of the influence of specific ghost substructures, we calculated the frequencies of each ghost substructure, in the actives and inactives of the top 1% ranked compounds. Using the 5HT reuptake inhibitors activity class as the example here, 46 active compounds and 984 inactives were identified by the HS 90 search. Ghost substructures were treated as duplicates if their InChI strings were identical (in other words, they are isomorphic graphs). We appreciate that this ignored the fact that isomorphic ghosts may arise from different bonds of the hyperstructure, but this purpose is primarily for visualisation (and is difficult to visualise dense graphs like hyperstructures). The most frequent ghosts are shown in Figure 5.5. There is a certain degree of overlap between the top five active and inactive ghosts identified, noting that two of the substructures are isomorphic, and the others exhibiting a similar size and topology. For the active ghosts, the mean frequency is 2.14 with a standard deviation of 2.83, whereas for inactives the mean frequency is 4.85 with a standard deviation of 21.3. In both cases the median frequency is 1, highlighting that the vast majority of ghosts identified occurred only once. The ghosts shown to have the greatest actives-to-inactives frequency ratio (Figure 5.5c) are somewhat different between the other two sets - whilst two of the ghosts occurred in the top five for actives and inactives, the other three are generally more dissimilar. This provides some indication that ghost substructures are indeed important for the retrieval of certain active compounds (presumably ones of different structural type to the actives used to build the compounds).



(a) Five most commonly occurring ghost substructures in the retrieved active compounds.



(b) Five most commonly occurring ghost substructures in the retrieved inactive compounds.



(c) Five ghost substructures which occur more commonly in actives than inactives. Determined on the basis of standardising their frequencies across all the substructures identified in the actives (and likewise for inactives), and measuring the difference in standardised frequencies.

Figure 5.5. Commonly occurring ghost substructures in the MDDR 5HT reuptake inhibitors activity class, resulting from the top 1% ranked compounds of the HS 90 search.

5.4 Conclusions

This chapter demonstrates, for the hyperstructure search technique used, that the W3 weighting scheme was competitive (if not slightly better) with W1, particularly for the hyperstructure-fingerprint fusion technique. However, as the differences in performance were only slight, and with fingerprints still generally outperforming the hyperstructure searches, this yields a result of otherwise little benefit in virtual screening. This was observed for active compound recall, yet no significant differences were observed for scaffold retrieval, suggesting that different proportions of analogues were obtained. Topology type matching in hyperstructures, surprisingly, had no beneficial effect on virtual screening, with little effect on the rankings of methods and weighting schemes, and with none of the methods outperforming fingerprints alone.

It is clear that some ghost substructures clearly are important in the retrieval of active compounds, though as mentioned by N. Brown (2002), some “rogue” substructures may compromise the results. Further investigation (i.e. substructural analysis) is needed to discern what sort of ghost substructures occur frequently in active and inactive compounds and how these affect recall and scaffold hopping. It may be worth applying substructural analysis to ghost substructures that have a disproportionate presence in active and inactive compounds, to observe influences the virtual screening performance of the current hyperstructure searches.

When compared with the results of this chapter, the work of Klinger and Austin (2006) suggests that the type of MCS, and MCS algorithm employed, have a significant effect on hyperstructure virtual screening (and ghost substructures). The effects on virtual screening performance will be explored in the next chapter, where different MCS algorithms and different MCS types are compared.

Chapter 6

Evaluation of MCS algorithms for Alignment and Virtual Screening

6.1 Introduction

Recently there has been a lack of extensive comparisons of MCS algorithms in chemoinformatics. Brint and Willett (1987) were the first authors to compare MCS algorithms, where a clique detection algorithm was compared with a subgraph enumeration algorithm to find the 3D MCS. Gardiner et al. (1997) reviewed a set of clique-detection algorithms for finding alignments of 3D molecules (a generally simpler problem compared to a 2D perspective). They identified two clique-detection algorithms worthy of note - the Bron-Kerbosch algorithm (Bron & Kerbosch, 1973), and the Carraghan-Pardalos algorithm (Carraghan & Pardalos, 1990). Conte, Foggia, and Vento (2007) more recently performed a comparison of three MCS algorithms, though these algorithms were published before the turn of the millenium and no algorithm emerged as being generally superior in terms of time performance. It is not known whether the exactness of an MCS algorithm is linked to virtual screening ability in similarity searching. Furthermore, it is not well known how different types of MCS (i.e. cMCS, dMCS and tdMCS) compare in similarity searching. This work sets out to evaluate an assortment of MCS algorithm types, both in terms of time complexity, and suitability for chemical similarity searching and hyperstructure virtual screening. For more information on MCS algorithms, readers are directed to chapter 2, as well as to reviews by Raymond and Willett (2002b) and Ehrlich and Rarey (2011).

6.2 Materials and Methods

6.2.1 A Note on Implementation

It is worth mentioning before going into depth with the methodology, that it is almost impossible to obtain similar results to those that the original authors presented for their algorithms. This is mainly due to differences in programming and implementations (i.e. some concepts explained in the papers have multiple methods of implementation). We have tried to implement the algorithms here as closely to the original papers as possible, but we appreciate that it is impossible to recreate the original implementation unless the source code and/or program is provided (as is the case with CDK's VFLibMCS algorithm). Of note is that all algorithms here have been programmed using Java for the KNIME platform (for details on hardware and software see the Methods section in Chapter 4), whereas most of the algorithms were originally implemented in C/C++, the latter of which will often result in faster run times. This chapter describes how we have implemented the aspects of the MCS algorithms, as well as the tests and evaluation measures used.

The MCS algorithms used in this paper are summarised in Table 6.1. All clique detection algorithms, and fMCS have been implemented with a time-out condition - that is, none of their algorithms may explore a given solution beyond this time limit. SMSD and VFLibMCS have also had the same time-out condition introduced. For the kcombu, consR and CDKMCS algorithms, there was no need to introduce a time-out condition due to their inexact nature, and thus would be highly unlikely to exceed the time constraints imposed in this chapter. ChemAxon_MCS has the same situation, though we did not possess the source code for this proprietary algorithm, and thus could not introduce a time constraint anyway.

Of the exact clique detection methods implemented in this work, we only found a cMCS solution for the Bron-Kerbosch algorithm, as depicted in Algorithm 3 of (Hariharan et al., 2011). In brief, this modification relied on expanding a clique with a given vertex, provided that said vertex possessed at least one c-edge (defined in Chapter 2) to a member of the current clique in question. Unfortunately we could not apply such a modification to the other three exact algorithms due to the heuristics involved in each algorithm. The algorithms that relied on graph colouring in particular caused problems when attempting to implement this condition. The c-clique problem (finding the cMCS via clique detection) to our knowledge cannot be solved by modifying the modular product alone. In fact, if a modular product is constructed purely of c-edges, then finding the cMCS is actually a case of finding the longest path, rather than being a clique detection problem. For these reasons, and since no cMCS

Output	MCS type	Algorithm	Reference(s)	Number of MCSs
Exact	cMCS	VFLibMCS	(Rahman et al., 2009)	>1
		SMSD	(Rahman et al., 2009)	>1
		fMCS	(Dalke, 2013)	1
All	All	MaxCliqueSeq (D)	(Depolli et al., 2013)	1
		Bron-Kerbosch (BK) *	(Bron & Kerbosch, 1973; Hariharan et al., 2011)	All
		Carraghan-Pardalos (CP)	(Carraghan & Pardalos, 1990)	1
Approximate	cMCS	RASCAL	(Raymond et al., 2002b)	1
		CDKMCS	(Rahman et al., 2009)	>1
		kcombu *	(Kawabata, 2011)	K (arbitrary parameter)
dMCS	dMCS	consR	(Zhu et al., 2011)	1
		ChemAxon_MCS (CA_MCS) *	(Englert & Kovács, 2015; Grosso et al., 2008)	1

Table 6.1

The MCS algorithms to be evaluated. "All" for MCS type refers to the ability of an algorithm to support dMCS, tdMCS and hMCS types. Abbreviations are referred to in brackets, which are sometimes used in the results tables due to space constraints.

* These algorithms can also be used to compute the cMCS.

solutions currently exist for the CP, RASCAL and MaxCliqueSeq algorithms, we have been unable to use them to calculate the cMCS in this study.

6.2.2 Modular Product of Two Graphs

As explained in detail in Chapter 2, the modular product of two graphs is sought using clique detection algorithms, where a maximum clique corresponds to a maximum common substructure. As we are seeking the MCES (edge-matching) instead of the MCIS (effectively, node matching), the modular product is constructed between the line graphs of the molecules, rather than the molecular graphs. Therefore, the node set $V(G)$ in the modular product G , represents the pairs of matched bonds between the molecular graphs, not the pairs of matched atoms.

The node set of the modular product in this work, $V(G)$, is defined as

$$V(G) = \bigcup_{i=1}^{|E(G_1)|} \bigcup_{j=1}^{|E(G_2)|} (u_i, v_j) : u_i \in E(G_1) \wedge v_j \in E(G_2) \wedge matches(u_i, v_j)$$

where G_1 and G_2 are two molecular graphs, $E(G_1)$ is the set of edges of G_1 , and *matches* is a function that checks if the weights of the two edges are compatible (i.e. bond order), and whether a bijective

mapping of the node labels of the two edges exists (or more ambiguously, the labels of the atoms match).

The edge set of the modular product, $E(G)$, is defined as

$$E(G) = \bigcup_{i=1}^{|V(G_1)|} \bigcup_{j=1}^{|V(G_2)|} (u_i, v_j) : \left(adj(u_{i1}, v_{j1}) \wedge adj(u_{i2}, v_{j2}) \right) \vee \left(\neg adj(u_{i1}, v_{j1}) \wedge \neg adj(u_{i2}, v_{j2}) \right)$$

where adj is a function which tests if a common vertex exists between two edges in the relevant molecular graph (G_1 or G_2). If a common vertex exists for each of the two specified modular product nodes in their respective molecular graphs, then the common vertices must be compatible (i.e. have the same label).

When constructing the tdMCS (as detailed in Chapter 2), the topological distance difference constraint is applied when building edges in the modular product, using a topological distance limit of θ . The Floyd-Warshall algorithm (Floyd, 1962) is used to calculate minimum path distances in molecules (between edges). If a pair of modular product nodes (or molecule edge pairs) has a topological distance difference that exceeds θ , then no edge is created between the two nodes.

In implementations here, unless otherwise explained, modular products have been represented as bitstring adjacency matrices. This is primarily inspired by the work of Depolli et al. (2013), due to the extensive requirement for setwise AND and OR operations in the clique detection algorithms, for which computer processors are well suited. This has not been used in Carraghan-Pardalos' algorithm, where an integer-based adjacency matrix is used instead. Whilst we could have re-designed this algorithm to use bitstrings, we felt that this would have deviated too far from the original specification of the algorithm.

6.2.3 Modular Product Simplification Heuristics

The node and edge “deletion” heuristics described by Raymond et al. (2002a) have been implemented, with a few changes to further improve search times. The rationale for implementing said heuristics was to improve MCS algorithm search speed (for methods which use the modular product), whilst minimising reductions in the size of the MCS obtained. Although referred to as “deletion” heuristics, it is better to think of these heuristics as additional rules for determining the creation of nodes and edges in the modular product. From here on, an MCS algorithm which uses these heuristics will retrieve a dMCS referred to as *hMCS* (*heuristic*-MCS). Although the same heuristics can be applied

with other MCS types (such as the tdMCS), we have only applied the heuristics for simplifying the dMCS problem. New edge heuristics for creating the hMCS, which were not featured in the original heuristic set (Raymond et al., 2002a), are introduced here. These new additions all utilise topological distance constraints when considering edge creation, the specific details of which can be found in Appendix H.

6.2.4 Compound Datasets

A number of compound sets were designed which we believe serve as informative benchmarks for the MCS algorithms presented here. Of main interest concerning this study is the time taken to retrieve a given (maximum) common substructure, as well as the number of bonds in the common substructure. Factors known to influence search time include the sizes of graphs being compared, symmetry, and the average degree of the graphs. The latter point is worth mentioning in light of the clique detection algorithms, where the modular product graphs created are particularly dense - owing to the low average degree in 2D chemical graphs.

The first set of compounds consists of symmetric compounds, which was chosen for testing in light of the remarks on symmetric compounds by Raymond et al. (2002a), who showed that the amount of symmetry in graphs being compared was related to the time taken to calculate the MCS. This set is shown in Table 6.2, the bonds in bold belonging to the MCS between each compound pair. S1 and S2 are examples that were historically used to test RASCAL. The next three pairs are somewhat flexible molecules where the first and second compounds are of similar sizes. These three pairs were introduced to test the performance of the flexible bond-based modular heuristics. The last example, S6, is being used due to the highly aromatic and inflexible nature of the compounds.

The next set of compounds in Table 6.3 has been chosen on the basis of having no rings. This is mainly to check the performance of the non-ring based heuristics implemented here. Notably the MCSs are rather fragmented, thus topological distance-dependent heuristics are not expected to work well. It is worth mentioning that the compounds in N1 are symmetric, yet the reported MCS is not (the reported MCS was retrieved by ChemAxon's MCS algorithm, which evidently found a mapping larger than or equivalent in size to a symmetric mapping).

A miscellaneous set is presented in Table 6.4. The first two compounds have been selected on the basis of being topologically non-planar. Non-planar graphs are those which cannot be drawn in 2D without causing edges to overlap. Although non-planar chemicals are rare occurrences, several

algorithms make assumptions about the planarity of graphs, including isomorphism, automorphism (Faulon, 1998) and subgraph isomorphism (Adler, Dorn, Fomin, Sau, & Thilikos, 2010). It would be interesting to see if non-planar molecular graphs yield different performance statistics from planar graphs (Dalke, 2012). The final case, M3, has been selected as a stress test to see how well the algorithms handle large rigid molecular graphs.

The next dataset was designed by Franco, Porta, Holliday, and Willett (2014) to yield one hundred pairs of compounds with differing degrees of similarity. The original purpose of this dataset was to assess the abilities of experts (at the European Medicines Agency), in judging chemical similarity. We have selected a subset of the 100 compound pairs, where every fourth compound has been chosen, after sorting the 100 pairs in descending order of RDKit Morgan Fingerprint similarity, at a radius of 2. From here, this fingerprint is referred to as mECPF4. These compounds are shown in Table 6.5.

The final benchmark concerns the use of the MCS in virtual screening. This experiment has been designed to address three questions:

- Whether the use of an exact algorithm is necessary in MCS similarity searching
- What influence the type of MCS has (i.e. dMCS, cMCS and tdMCS)
- How MCS group fusion compares to fingerprint group fusion.

We focus on a small selection of the ChEMBL compound sets in Riniker and Landrum (2013). The ChEMBL datasets here are interesting for virtual screening as each activity class has 100 active compounds, yet a single set of 10 000 inactives is re-used for each activity class. The classes also possess a wide range of intra-set similarities and, more importantly, have highly variable performances on fingerprint searches. The subset focused on is shown in Table 6.6. These classes have been chosen on the basis of the authors' reported fingerprint performance. Two datasets with a high mean intra-set similarity (\bar{S}_{AA}) were noted to be high-recall classes for fingerprints (Somatostatin receptor 5 and C5a receptor). Two other classes with a low intra-set similarity and a high number of unique frameworks have also been chosen (Cytochrome P450 2C9 and beta-2 adrenergic receptor). The authors noted that the AUC ROC values for these two classes were close, if not below, the random threshold of 0.5. The final two classes in the table, have been chosen on the basis of having intermediate virtual screening performances (from supplementary data in the author's publication, these two classes had BEDROC scores close to the median BEDROC value).

Number	Compound 1	Compound 2	Reference(s)
S1			(Raymond et al., 2002b)
S2			(Raymond et al., 2002a)
S3			(Ersmark et al., 2003)
S4			(Ersmark et al., 2003)
S5			(Bone, Vacca, Anderson, & Holloway, 1991)
S6			(Libby, Munson, Fiel, & Porter, 1995)

Table 6.2
 Pairs of symmetric compounds to be evaluated

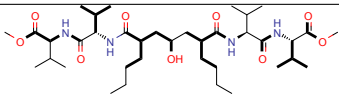
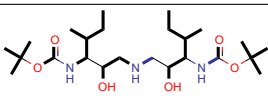
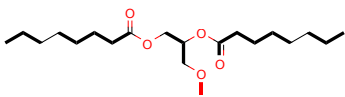
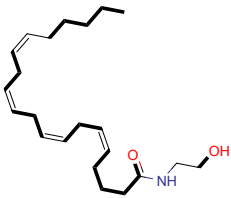
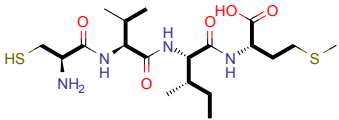
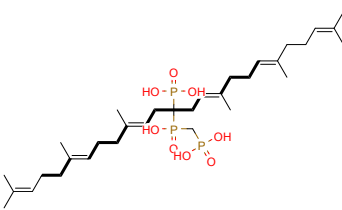
Number	Compound 1	Compound 2	ChEMBL IDs
N1			CHEMBL39130, CHEMBL46316
N2			CHEMBL26440, CHEMBL15848
N3			CHEMBL55423, CHEMBL1204752

Table 6.3
Pairs of non-ring compounds to be evaluated

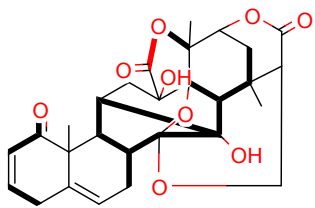
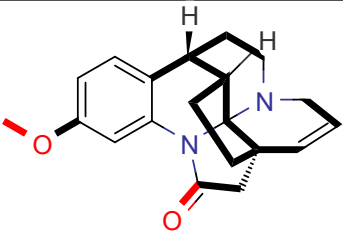
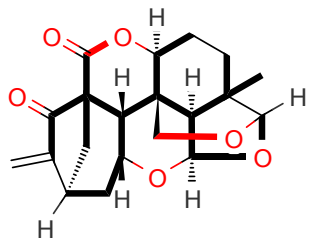
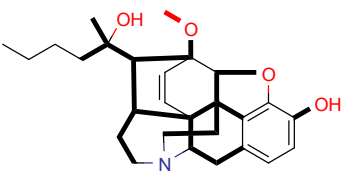
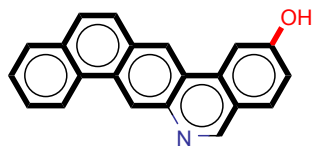
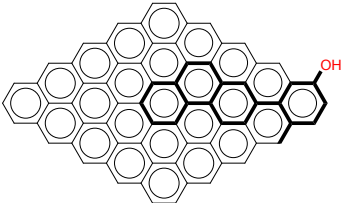
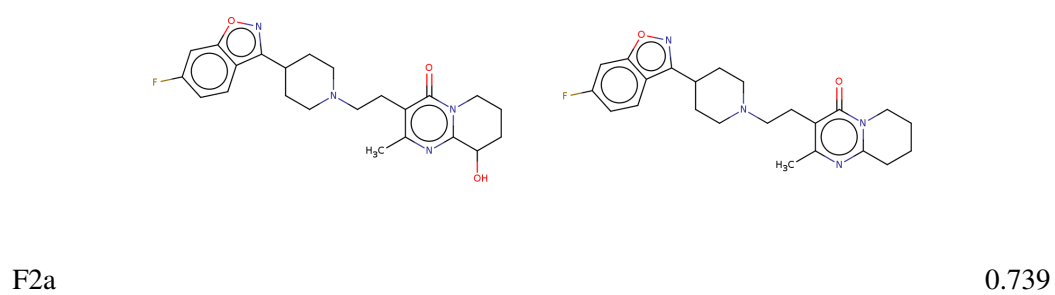
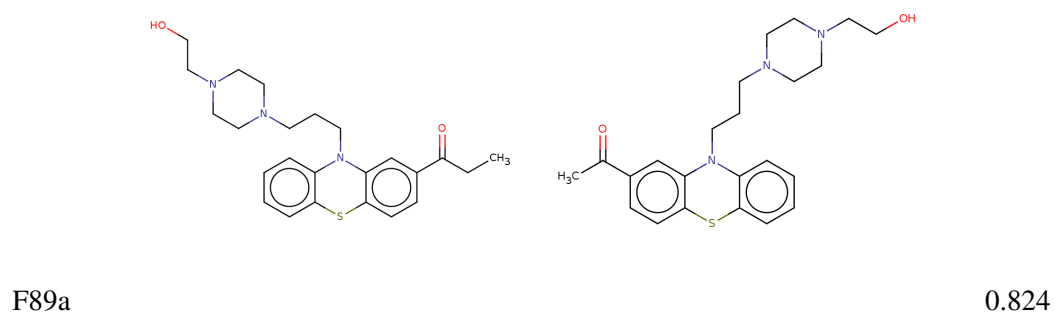
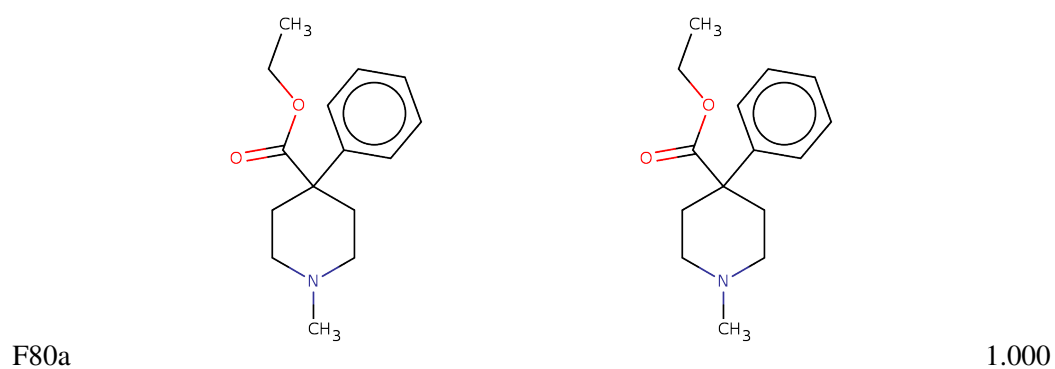
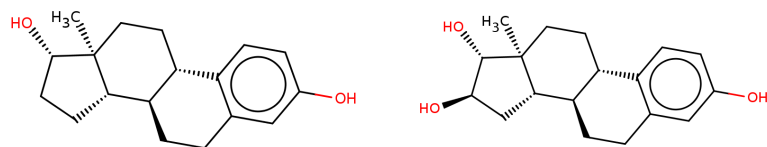
Number	Compound 1	Compound 2	References
M1			(Kariba, Houghton, & Yenesew, 2002), (C. Huang et al., 2014)
M2			(Li et al., 2007), (Maurer & Rapoport, 1987)
M3			N/A

Table 6.4

Pairs of miscellaneous compounds to be evaluated

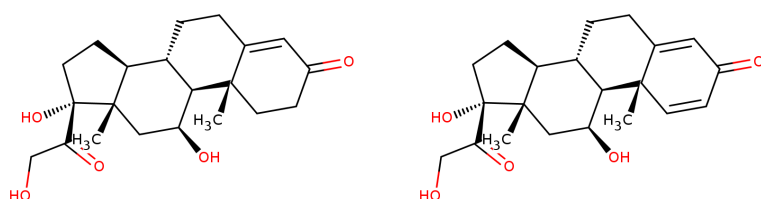
Number	Compound 1	Compound 2	Similarity
--------	------------	------------	------------





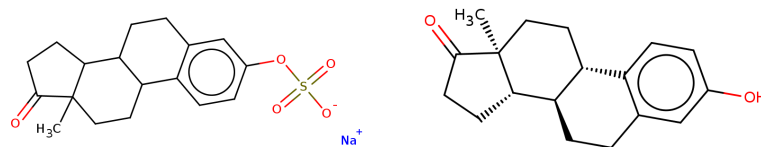
F57a

0.723



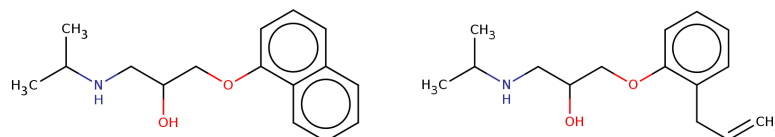
F92a

0.678



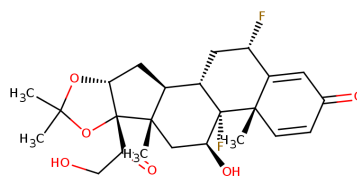
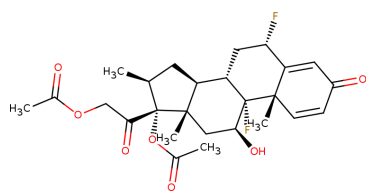
F31a

0.636



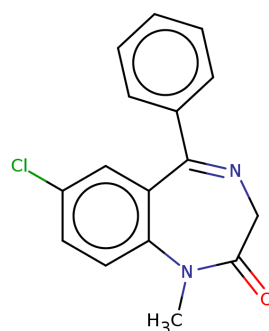
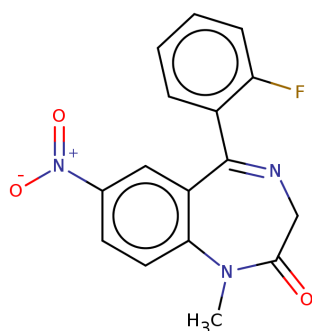
F19a

0.587



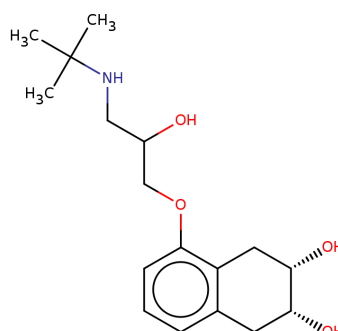
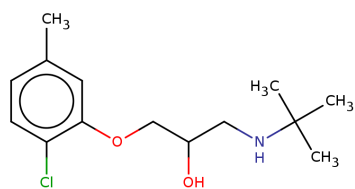
F100a

0.544



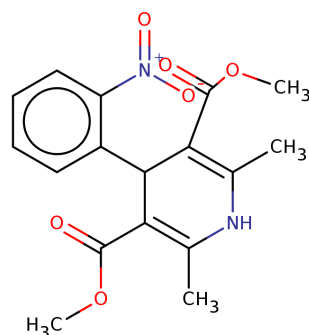
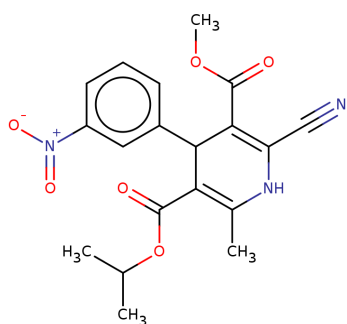
F91a

0.500



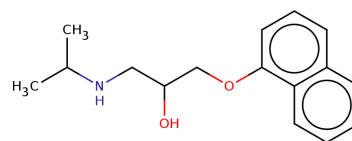
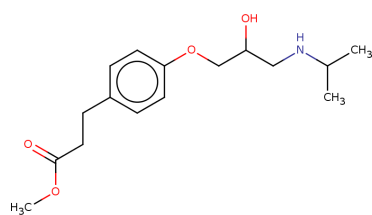
F94a

0.481



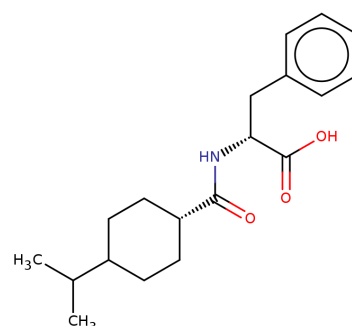
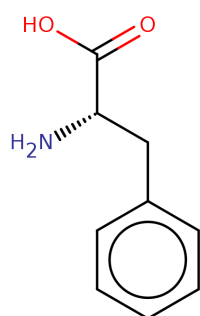
F72a

0.444



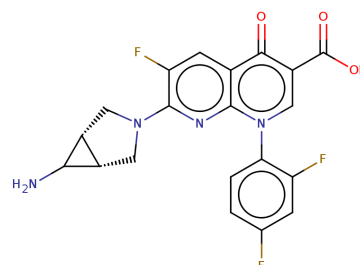
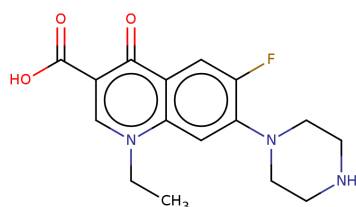
F18a

0.415



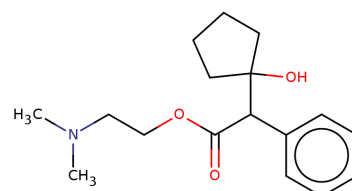
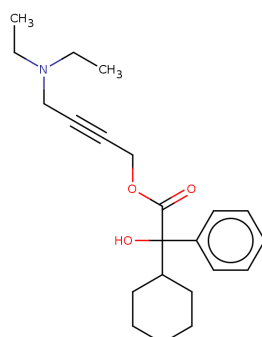
F34a

0.391



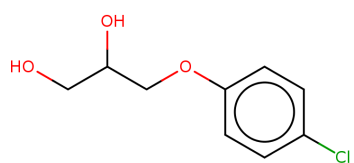
F47a

0.366

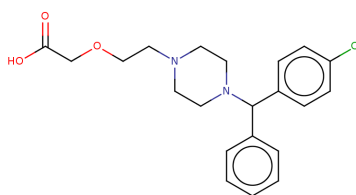


F69a

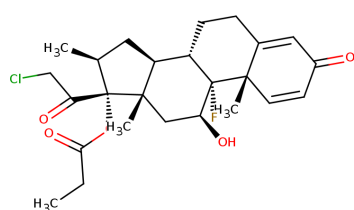
0.279



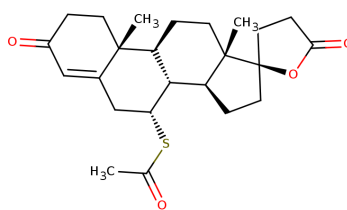
F28a



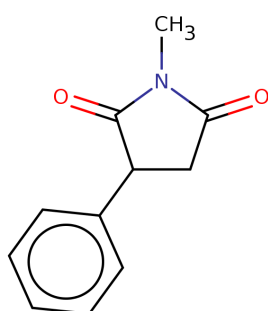
0.214



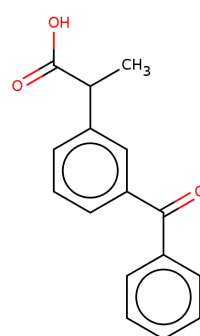
F96a



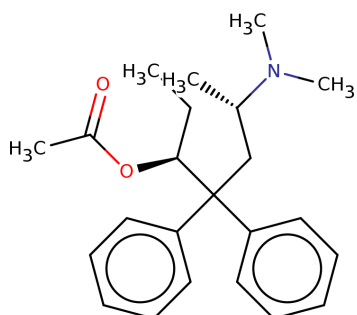
0.186



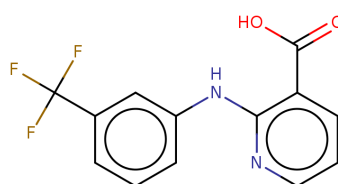
F50a



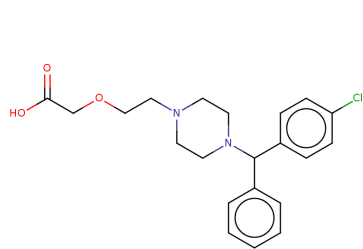
0.170



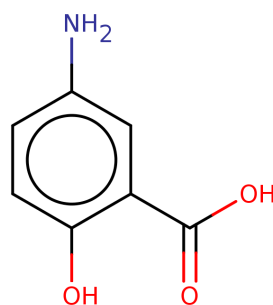
F8a



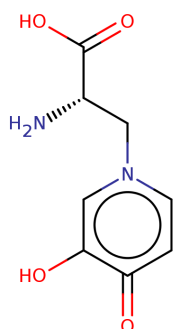
0.152



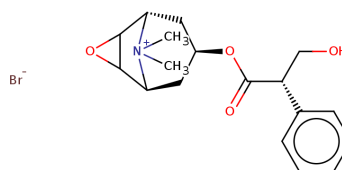
F13a



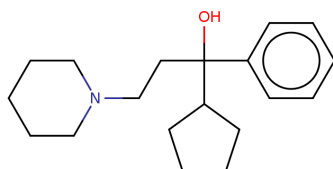
0.138



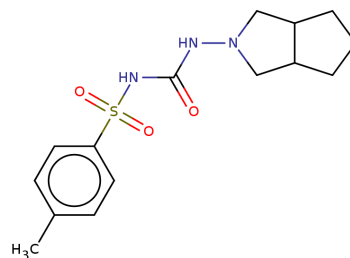
F45a



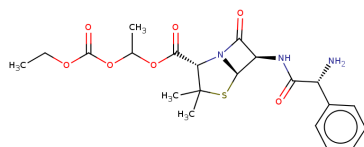
0.134



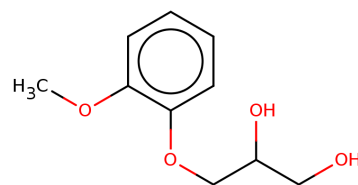
F12a



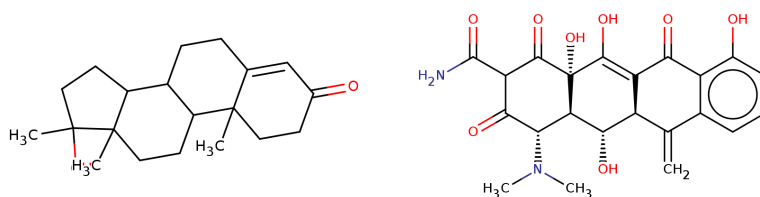
0.129



F30a

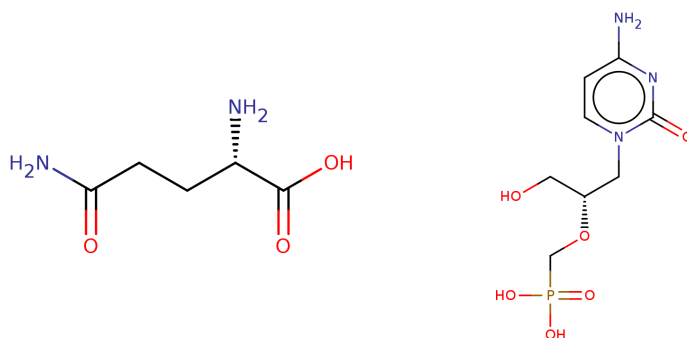


0.117



F20a

0.105



F88a

0.091

Table 6.5

Franco compound pairs to be evaluated, sorted in descending order of mECFP4 similarity (as depicted in the “Similarity” column). “Number” represents the name of the compound pair that the authors originally assigned.

ChEMBLID	Name	\bar{S}_{AA}	S_{AA}^{σ}	\bar{S}_{AI}	S_{AI}^{σ}	Frameworks
CHEMBL1792	Somatostatin receptor 5	0.268	0.067	0.144	0.021	59
CHEMBL3397	Cytochrome P450 2C9	0.120	0.014	0.125	0.021	96
CHEMBL2373	C5a anaphylatoxin chemotactic receptor	0.283	0.063	0.142	0.013	45
CHEMBL5414	beta-2 adrenergic receptor	0.137	0.020	0.132	0.020	94
CHEMBL1889	Vasopressin V1a receptor	0.168	0.023	0.135	0.014	87
CHEMBL1957	Insulin-like growth factor I receptor	0.162	0.026	0.128	0.015	86

Table 6.6

Similarity statistics for the classes used in the ChEMBL-derived datasets, using RDKit standard Morgan fingerprints at a radius of 2 as the descriptor. \bar{S} represents the mean pairwise similarity (MPS) and S^{σ} is the standard deviation of similarity values. Subscript AA denotes similarity between active molecules in the class, and AI is the inter-similarity between actives and inactives. “Frameworks” refers to the number of unique Bemis-Murcko frameworks as calculated with RDKit (as detailed in the methodology of Chapter 4).

6.2.5 Benchmarks

6.2.5.1 Size and Time Performance on Compound Pairs. Our tests on the compound pairs described (including the Franco set) are concerned with algorithm run-time and the size of the common subgraph retrieved. We are fundamentally testing four MCS types - dMCS, tdMCS, hMCS and cMCS. Details on what MCS type applies to a given algorithm are shown in Table 6.1. The algorithms in the “All” category of MCS type will be tested on the dMCS, hMCS and tdMCS (with θ values of 0 and 2).

Obtaining accurate time benchmarks using Java programs is a non-trivial task. Java is both a compiled and interpreted language, whereby the source code is compiled to Java byte code, which is then either interpreted, or compiled to machine code, by a Java Virtual Machine. Several optimisations are built into the compilation process in order to speed up program run time, which includes elements like dead code elimination and loop unrolling. In addition, not all the byte code is converted to machine code. The Java Just-In-Time compiler only compiles a code path when it knows that a given code path is about to be executed. More recently, only frequently-executed code paths are compiled; such dynamic compilation requires that the code is executed a number of times. Benchmarkers thus may attempt to “warm-up” their code by performing progressive executions and compilations in order to get the most out of this dynamic compilation process. However, this also means that benchmarking will often not reflect the actual time performance observed by the majority of users (Goetz, 2004, 2005). In this work we have not taken any steps to account for said compilation optimisations. Unfortunately, this can give rise to a large amount of error in measured time performances. This is why in this study, we have calculated median times (and associated statistics), instead of mean times, due to the high likelihood of outlying results.

For each pair of compounds, the MCS search was performed 10 times, and the median time and median absolute deviation (mad) was calculated. The method for calculating mad was the methodology as used in R, where for a given numeric set X ,

$$mad = median(|x - median(X) \forall x \in X|) \times 1.4826$$

the constant of 1.4826 being used to estimate the standard deviation (assuming that X is normally distributed).

To prevent excessive run times, a limit of 300 seconds has been applied for each algorithm. A run that exceeds said time limit is referred to as a “non-completion” for the rest of this chapter, as the

solution is not guaranteed to be the optimum for the algorithm. As a result of the cut-off, some exact algorithms will not yield exact solutions in the results.

6.2.5.2 Virtual Screening. For each activity class of the compounds from Riniker and Landrum (2013), 5 reference structures have been chosen to use for both mECFP4-based group fusion, and MCS-based group fusion using a subset of the described MCS algorithms (as described in the results and discussion). The group fusion in question was the application of the MAX rule on the similarity scores. The idea behind using a subset of MCS algorithms is to avoid algorithm redundancy (that is, to avoid algorithms which yield the same answer but at a worse time performance). The subset consists of:

- The fastest exact clique detection method, used to calculate hMCS, and tdMCS (θ ranging from 0 to 2 as tested by Kawabata (2011)). dMCS will be excluded on the basis that it is computationally too slow to calculate.
- The fastest exact cMCS method
- The three approximate dMCS methods, all of which are used (instead of the one that yields the largest MCS) due to their fundamentally different methods for obtaining the MCS.

The Tanimoto coefficient has been used to calculate similarity as used in Chapter 4. The BEDROC statistic with a value of 160.9 for α was used to assess the virtual screening ability of a method. We note that the weak and strong ring heuristics can yield some reductions in MCS size when comparing rings of different sizes (Figure H.4). Due to this potential problem, the hMCS was tested with the fastest clique detection algorithm separately; where the ring heuristics were either on, or off. As these tests involved the comparisons of thousands of pairs of molecules, we set the time cutoff to 10 seconds. This should be an adequate cutoff, noting that we ultimately want a technique that is not unfeasibly slower than fingerprints for large-scale similarity searching.

Algorithm	cMCS	dMCS	hMCS	tdMCS ($\theta = 0$)	tdMCS ($\theta = 2$)
VFLibMCS	K.5, L.5				
CDKMCS	K.5, L.5				
SMSD	K.5, L.5				
fMCS	K.5, L.5				
MaxCliqueSeq		K.1, L.1	K.2, L.2	K.4, L.4	K.3, L.3
BK	K.5, L.5	K.1, L.1	K.2, L.2	K.4, L.4	K.3, L.3
CP		K.1, L.1	K.2, L.2	K.4, L.4	K.3, L.3
RASCAL		K.1, L.1	K.2, L.2	K.4, L.4	K.3, L.3
kcombu	K.5, L.5	K.1, L.1	K.2, L.2	K.4, L.4	K.3, L.3
consR		K.1, L.1			
ChemAxon_MCS	K.5, L.5	K.1, L.1			

Table 6.7

References to raw data for an MCS algorithm. Each cell in the table points to the table of results relevant to the type-algorithm combination, which displays performances for individual compound pairs. For each cell, the first table identifier relates to results for the S, N and M compound pairs, whereas the second refers to the Franco set.

6.3 Results and Discussion

Due to the large number of tables of raw data shown in this chapter, Table 6.7 has been created to direct the reader to the relevant table(s) for a given algorithm. Statistics-derived figures and tables, however, are not referenced in this table.

6.3.1 S, N and M Compound Pairs

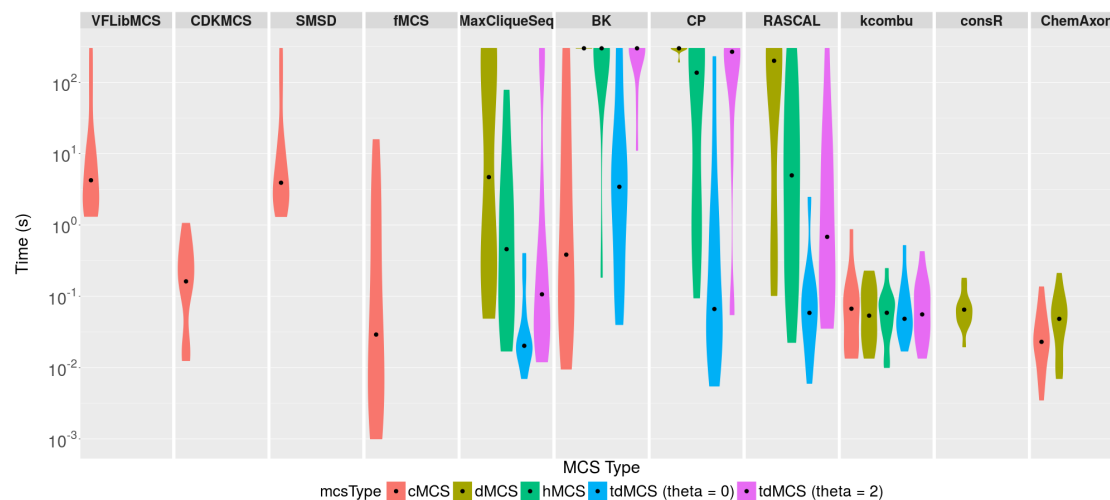
For the algorithms which have been tested on the dMCS, hMCS and tdMCS types (being BK, CP, RASCAL, MaxCliqueSeq and kcombu), median times and sizes are shown in 6.8, which details the influence of the MCS type on the search results, for a given algorithm. In this table, Kendall's W tests have been performed to quantify the agreement amongst algorithm rankings per compound pair. The W statistic and p-value are displayed in these tables, where a significant p value indicates that the mean ranks are informative for comparing relative algorithm and MCS type performance. Performance information for each pair of compounds is shown for the applicable algorithms for each MCS type, in Tables K.1, K.2, K.3, K.4 and K.5 (in Appendix K) for dMCS, hMCS, tdMCS ($\theta = 2$), tdMCS ($\theta = 0$) and cMCS respectively (where the terminology is explained in the first of these tables). Modular product size, density and construction time for each pair of compounds, for each MCS type (excluding cMCS) are shown in Table J.1.

The charts in Figure 6.1 show respectively for each MCS algorithm (and for each MCS type per algorithm where applicable):

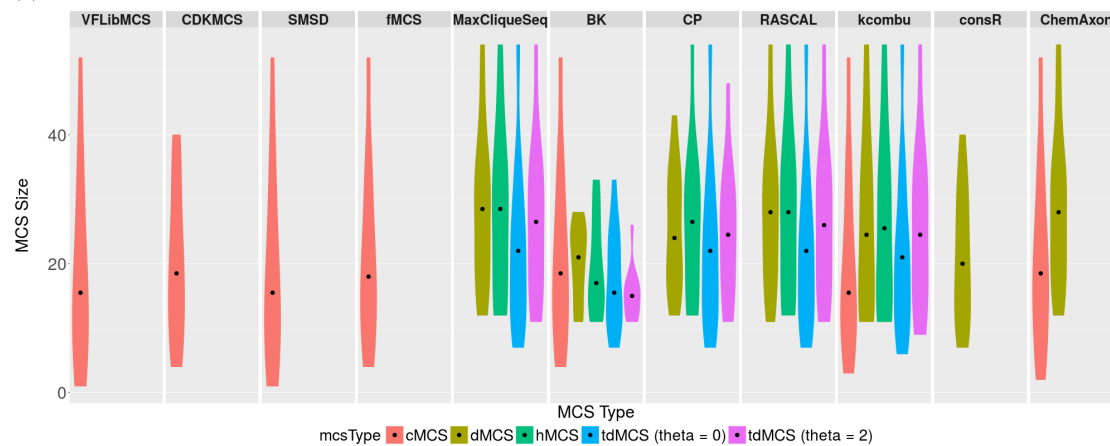
- A violin plot representing the time distributions for each algorithm-MCS type combination. A violin plot is an extension of the box-and-whisker plot, where each “violin” is a plot of the density function of a histogram produced from the data (Hintze & Nelson, 1998). This has some advantages over simple box plots, for it shows how the data are distributed (for example, a box plot would not distinguish a bimodal distribution from a uniform one). In this work, the maximum width for each “violin” is constant, the width representing the probability density at a given point. Maxima and minima of the compound times are represented by the corresponding extremes for each “violin.” The black dots represent the median of the data. The box-and-whisker plots have themselves been omitted from these violin plots as they complicate the visualisation, and will not add much useful information that the frequency distributions do not already show.
- A violin plot (as explained above) for MCS sizes
- A bar chart depicting the number of compound pairs for which a non-completion occurred.

In addition, a summary chart depicts the compromise between size and time performance for each method (Figure 6.2). The mean ranks in this plot are calculated using the medians of size or time, per compound pair.

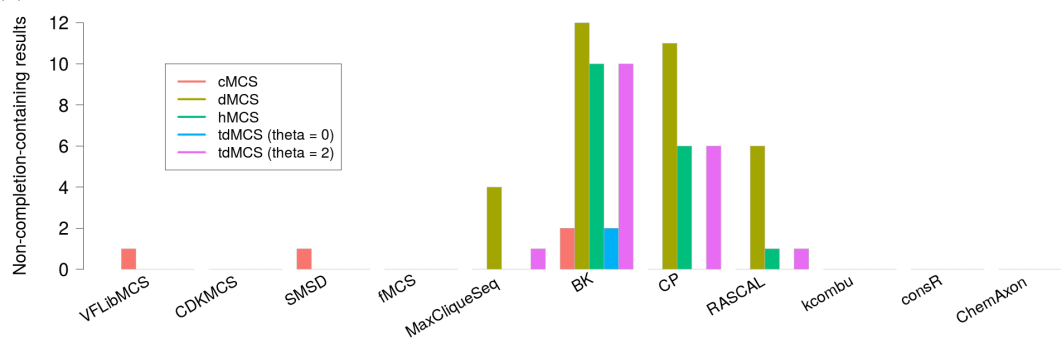
6.3.1.1 Modular Product Observations. It is expected that as the modular product is further simplified (in descending order of complexity: dMCS; hMCS; $\theta = 2$; $\theta = 0$) the performance of all clique-based algorithms improve in terms of time. Table 6.8 effectively summarises this point, where the median times also decrease in the mentioned order of complexity. The rankings of the median times were significant, as shown by the (column-wise) Kendall’s W statistics. It is clear that imposing topological constraints had an effect on the MCS search, seeing that the modular product density was reduced, comparing Tables K.1 and K.4 (yet the number of nodes remained unchanged). This means that practically all MCS searches ran quicker, though larger graphs were still hard to solve the MCS for, noting that when $\theta = 2$, larger compound pairs like M3, S5 and S6 were slow, even when using the faster performing algorithms. It is interesting to note that kcombu (an inexact algorithm) had different time performance trends. The Kendall’s W result (column-wise) is not significant for kcombu, implying that MCS type has little influence on kcombu’s search time, which is not surprising given that kcombu runs for a fixed number of iterations per MCS. From the violin plots, it can be seen that most of the time distributions are normally distributed. There are some exceptions - for instance,



(a) Time taken



(b) MCS size



(c) Number of non completions for a method.

Figure 6.1. Charts showing performance information of each algorithm (including the same algorithm ran on different MCS types) for S, N and M compound pairs.

the time performances from kcombu are more uniform than normal, and the dMCS distribution for MaxCliqueSeq being bimodal. It can also be observed, that the ranges of the distributions differ for each modular product type. Notably, tdMCS ($\theta = 0$) for MaxCliqueSeq and RASCAL has a more restricted range than for dMCS. This would be expected for all the exact algorithms, but due to the time constraints imposed for each compound pair, this is not properly observable in this work. By contrast, when looking at the time distributions for kcombu, the distributions become progressively more normal in descending order of modular product complexity.

When looking at MCS size, a similar descending order can be observed as was for size (dMCS; hMCS; $\theta = 2$; $\theta = 0$). Ideally, hMCS is supposed to have as little difference in size from dMCS as possible. The average size of hMCS was generally between the sizes of dMCS and tdMCS ($\theta = 2$). The significance of the algorithm performance results however (both in size and time) decreased as the modular product simplification increased. For example, with tdMCS ($\theta = 0$), there was no significant consensus in size rankings - implying that all algorithms had similar performances in MCS size for this MCS type. This was most likely due to the time cutoff not being exceeded, thus a greater proportion of exact solutions have been obtained. From the violin plots for size, a pattern can be observed where as the complexity of the modular product decreased, the size distributions become more skewed towards a shorter time, yet the ranges in size performance increase (the upper end reflecting compound pair S6, whose MCS size was the same for all modular product variations).

The effects that hMCS had are more complicated to explain. Looking at the compound classes (symmetric, non-ring and miscellaneous types), hMCS consistently improves the search speed (Compare Tables K.1 and K.2). This is partly because of the presence of ring systems which the heuristics act on, noting a particularly dramatic effect with S6 (where a sharp drop in modular product density is observed, as shown in Table J.1). An effect is even observed in S5, where despite the pair of molecules having different longest chain sizes (a large difference in longest chain sizes will reduce the effect of the chain topological distance difference heuristic), a reduction in time taken still occurred in all the algorithms which ran to completion. Unsurprisingly, there was a smaller improvement in performance associated with the N compounds, noting for a start that no nodes were deleted in their modular products by the heuristics. The C-C and non-ring induced subgraphs, as well as the chain topological distance difference heuristic, were the heuristics which acted on these compound pairs, due to the absence of rings and dominance of rotatable bonds. Note that the sizes of N1 and N2 MCSs in the hMCS results are smaller than for dMCS as a result of these heuristics. It is disappointing to see that whilst the non-planar compounds (M1 and M2) had large speed improvements, the sizes of

the hMCS were smaller than the dMCS equivalents. More work needs to be done to tune the ring and rigid-bond based heuristics to account for such compounds. The hMCS of M3 however, was achieved quickly with no difference in size between dMCS and hMCS.

6.3.1.2 Algorithm Observations. On looking at time results, BK was the slowest algorithm in all MCS cases tested here, an unsurprising result considering that BK is supposed to retrieve all maximal cliques, rather than one maximum clique. The only test case where it actually completed in less than 300 seconds most of the time was when $\theta = 0$, where it still failed in 2 cases. By contrast, ChemAxon's algorithm (dMCS) generally outperformed all others in terms of size, and to a slightly lesser extent was superior in time as well (Figure 6.2). A Wilcoxon signed-rank test comparing the results of ChemAxon_MCS and MaxCliqueSeq (dMCS) showed that the former was faster ($p < 0.05$). This speed result is impressive given the difficulty of the compounds, though unsurprising considering the fixed number of iterations (and thus, relatively fixed maximum runtime) of this algorithm. This implies that the ChemAxon heuristics used are very important in guiding the MCS search. RASCAL and MaxCliqueSeq were the fastest of the exact "All" algorithms. MaxCliqueSeq almost always outperformed RASCAL in both size and time, even on dMCS. Generally CP was the slowest maximum (not maximal) clique algorithm, noting that for dMCS it only obtained a complete solution for one compound pair (BK failing for all 12 pairs). Interestingly, CP seemed to somewhat outperform RASCAL for some pairs when calculating the tdMCS ($\theta = 0$), possibly as each iteration of RASCAL requires more computation time than CP. One would infer from this observation, that RASCAL normally needs less iterations than CP to converge (though this has not been tested in this study), though RASCAL generally remains the faster of the two algorithms, based on median time. Generally, kcombu was the fastest performing of the five "All" algorithms; its time not exceeding one second per comparison in this dataset (an unsurprising result given its inexact nature). consR by contrast, was the slowest of all the approximate dMCS algorithms. Looking at the time distributions (violin plots) for each algorithm, it can generally be said that the inexact algorithms have shorter ranged distributions than the exact ones. No other apparent patterns however, have been found.

Amongst the cMCS algorithms, fMCS performed considerably faster than the the SMSD and VFLibMCS cMCS algorithms. CDKMCS was the fastest of all the CDK algorithms (the other CDK algorithms being VFLibMCS and SMSD), and was somewhat competitive with fMCS and ChemAxon_MCS on cMCS mode (generally performing slower, though with some exceptions from individual compound pairs). kcombu (cMCS and other MCS types) was slightly slower in speed to fMCS. ChemAxon_MCS on cMCS mode however, was the fastest cMCS method in general. BK

(cMCS) had an intermediate median time performance of the cMCS algorithms, though like fMCS it had a considerably large range and spread of times. ChemAxon's algorithm by contrast, with its comparatively small range and deviation was evidently the least compound-dependent (and otherwise most robust) method when concerning time, with CDKMCS not far behind.

BK was the worst-performing algorithm when $\theta \neq 0$ (Figure 6.1), in terms of the size of MCS retrieved. CP was the second worst in terms of size for the "All" category of algorithms. MaxCliqueSeq and RASCAL by contrast generally obtained the largest MCS (for dMCS, tdMCS and hMCS). consR, despite being a dMCS algorithm, was generally outperformed by all the other dMCS algorithms in terms of size. ChemAxon_MCS on dMCS mode, retrieved the largest dMCS of all the inexact methods, in many cases obtaining equivalent if not larger common substructures than the best results from the exact methods ($p > 0.05$ with a Wilcoxon signed-rank test compared with size results from MaxCliqueSeq ran for dMCS). An exception is the compound pair N1, where MaxCliqueSeq retrieved a larger dMCS result than Chemaxon_MCS did. It is noteworthy that the size performance of kcombu was often better than BK and CP, and often retrieved similar sizes to RASCAL and MaxCliqueSeq in hMCS. However, it was still outperformed by ChemAxon_MCS.

Despite being inexact, CDKMCS had little difference in MCS size from the other two CDK algorithms, and thus represents the method of choice for time when using only CDK algorithms. The compound pair S6 however was an exception (CDKMCS cMCS yielded 40 bonds, whereas the largest cMCS is actually 52), which explains the lower maximum for CDKMCS than the other cMCS algorithms. VFLibMCS and SMSD of note, failed to find a solution for M3, which is why the reported number of bonds is 0 for that pair. BK and fMCS both were the best-performing algorithms in terms of size. Being an approximate method, kcombu (on cMCS mode) retrieved slightly smaller common subgraphs than the exact cMCS algorithms. ChemAxon_MCS on cMCS mode was the fastest cMCS method, though in a number of cases in the individual compound pairs, it falls short of the sizes retrieved by fMCS. fMCS is therefore the method of choice for cMCS searching in this work, for it was the fastest of the exact methods, and had a comparable time performance with the inexact cMCS methods overall.

6.3.2 Franco Compounds

Tables L.1, L.2, L.3, L.4 and L.5 show performance information for dMCS, hMCS, tdMCS ($\theta = 2$), tdMCS ($\theta = 0$) and cMCS respectively (in Appendix L). Median times and sizes for the "All" category algorithms are shown in 6.9, along with W statistics as mentioned in the previous section. Summary

row ID	BK	CP	kcombu	MaxCliqueSeq	RASCAL	W	p
dMCS	> 300	> 300	$5.65 \cdot 10^{-2}$	$1.19 \cdot 10^1$	$2.17 \cdot 10^2$	0.85	$3.29 \cdot 10^{-8}$
hMCS	> 300	$1.81 \cdot 10^2$	$6.10 \cdot 10^{-2}$	$6.50 \cdot 10^{-1}$	$4.96 \cdot 10^0$	0.86	$2.45 \cdot 10^{-8}$
tdMCS ($\theta = 0$)	$3.43 \cdot 10^0$	$6.73 \cdot 10^{-2}$	$4.85 \cdot 10^{-2}$	$2.02 \cdot 10^{-2}$	$5.97 \cdot 10^{-2}$	0.69	$1.02 \cdot 10^{-6}$
tdMCS ($\theta = 2$)	> 300	$2.70 \cdot 10^2$	$5.73 \cdot 10^{-2}$	$1.41 \cdot 10^{-1}$	$7.60 \cdot 10^{-1}$	0.86	$2.14 \cdot 10^{-8}$
W	0.73	0.81	0.05	0.76	0.87		
p	$8.37 \cdot 10^{-6}$	$2.07 \cdot 10^{-6}$	$6.36 \cdot 10^{-1}$	$4.85 \cdot 10^{-6}$	$7.73 \cdot 10^{-7}$		

(a) Median search times (in seconds)

row ID	BK	CP	kcombu	MaxCliqueSeq	RASCAL	W	p
dMCS	21.0	24.0	24.5	28.5	28.0	0.58	$1.34 \cdot 10^{-5}$
hMCS	17.0	26.5	25.5	28.5	28.0	0.42	$4.51 \cdot 10^{-4}$
tdMCS ($\theta = 0$)	15.5	22.0	21.0	22.0	22.0	0.26	$1.45 \cdot 10^{-2}$
tdMCS ($\theta = 2$)	15.0	24.5	24.5	26.5	26.0	0.57	$1.60 \cdot 10^{-5}$
W	0.18	0.18	0.45	0.50	0.57		
p	$9.32 \cdot 10^{-2}$	$9.54 \cdot 10^{-2}$	$1.01 \cdot 10^{-3}$	$4.19 \cdot 10^{-4}$	$1.25 \cdot 10^{-4}$		

(b) Median MCS sizes

Table 6.8

Comparisons between MCS types (or modular product modifications) for the S, N and M compound pairs, where an algorithm has results for dMCS, hMCS and tdMCS types. Kendall's W information row-wise, shows the agreement between ranks of MCS algorithms that the judges (different compounds for a given MCS type) have assigned. There are 12 judges (being 12 compounds in this case) and 5 objects (the MCS algorithms). Column-wise W information by contrast, shows the reverse information. Thus, there are 5 judges (MCS algorithms) and 12 objects (different compounds for an MCS type).

statistics for all the algorithms for each applicable MCS type are displayed in Figure 6.3. Modular product information can be seen in Table J.2.

In general, the patterns observed in Figure 6.3 reflect those observed in the previous dataset. The time performances were in general faster than the S, N and M pairs, usually with less non-completions occurring. There is little difference in performance patterns compared to the other set of compounds, for both approximate and exact algorithms. In this dataset, the searches were very fast, as this is a more trivial compound set (smaller compounds with less flexible bonds and symmetry) than the S, M and N benchmark set. As this set was more trivial to search than the S, M and N set, it can be more clearly observed that as the complexity of the modular product decreases, the distributions for the exact algorithms' time performances tend to have smaller ranges.

It is noteworthy that the Kendall's W values for MCS size were consistently lower in this set than in the S, M and N compounds, particularly for the W test results comparing algorithm ranks (as opposed

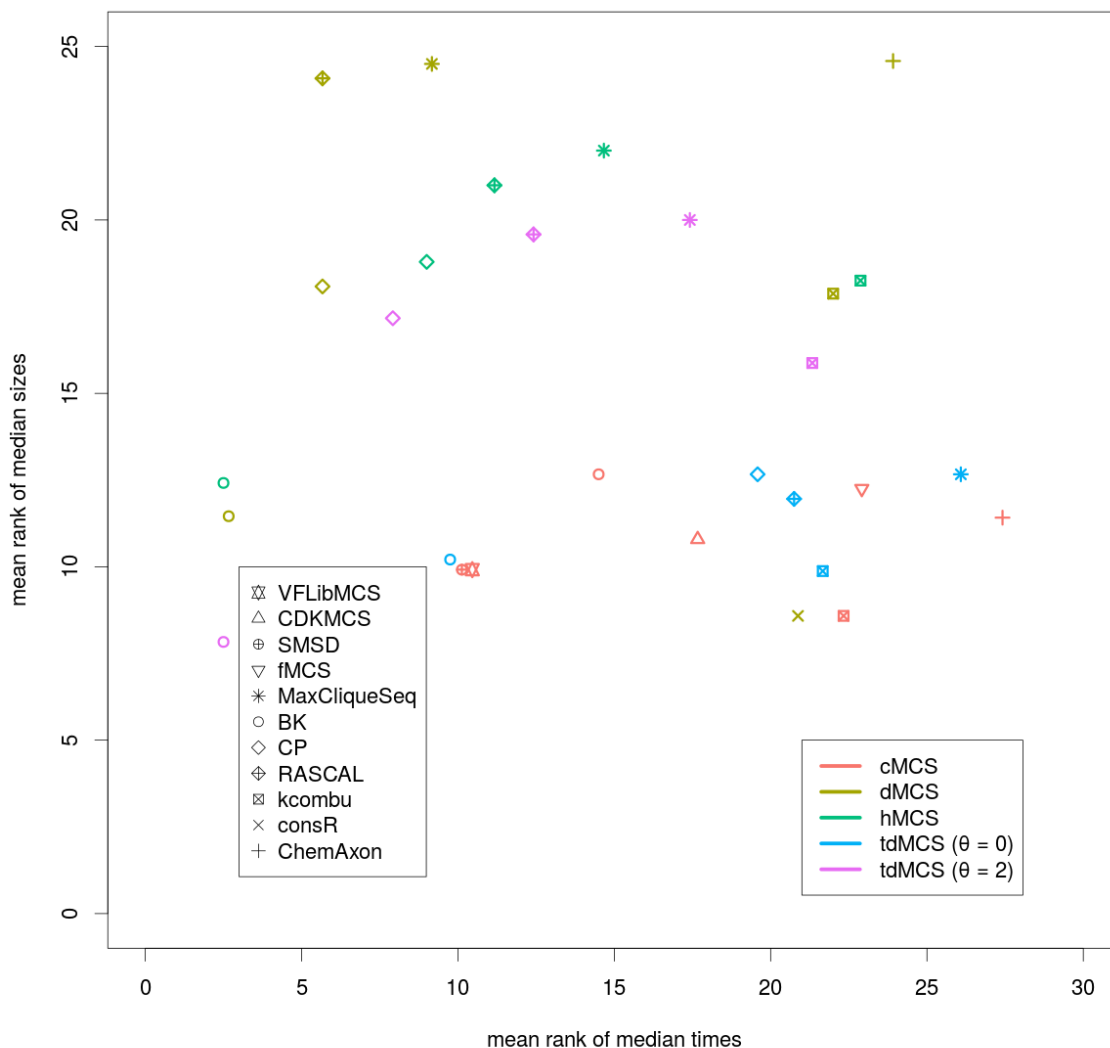


Figure 6.2. Plot showing algorithm-MCS types comparing mean size ranks and time, over each of the S, N and M pairs. A larger rank implies better performance (so lower time and larger MCS size).

to MCS type influence). This implies that for this dataset, there was less of a benefit in using a more powerful algorithm. Apart from tdMCS ($\theta = 0$) however the rankings were still significant. For both time and size performance, MaxCliqueSeq was once again consistently superior to other exact clique detection algorithms in terms of size and time. Unlike in the S, N and M set, MaxCliqueSeq actually outperformed ChemAxon_MCS in terms of size, though in dMCS mode was still outperformed for time. The CP algorithm interestingly has an improved profile compared to the S, M and N compounds, nearing the other two high-performance algorithms in size and time performance (although it has more non-completions). In particular, CP outperformed all the other applicable algorithms in

row ID	BK	CP	kcombu	MaxCliqueSeq	RASCAL	W	p
dMCS	> 300	$6.45 \cdot 10^{-1}$	$1.00 \cdot 10^{-2}$	$7.00 \cdot 10^{-3}$	$3.65 \cdot 10^{-2}$	0.75	$1.64 \cdot 10^{-15}$
hMCS	$1.53 \cdot 10^0$	$1.40 \cdot 10^{-2}$	$9.00 \cdot 10^{-3}$	$4.50 \cdot 10^{-3}$	$1.30 \cdot 10^{-2}$	0.60	$2.79 \cdot 10^{-12}$
tdMCS ($\theta = 0$)	$4.15 \cdot 10^{-2}$	$3.00 \cdot 10^{-3}$	$1.15 \cdot 10^{-2}$	$3.00 \cdot 10^{-3}$	$4.00 \cdot 10^{-3}$	0.77	$6.78 \cdot 10^{-16}$
tdMCS ($\theta = 2$)	$5.22 \cdot 10^0$	$1.40 \cdot 10^{-2}$	$1.75 \cdot 10^{-2}$	$5.00 \cdot 10^{-3}$	$1.55 \cdot 10^{-2}$	0.55	$3.52 \cdot 10^{-11}$
W	0.75	0.75	0.30	0.66	0.79		
p	$3.13 \cdot 10^{-12}$	$3.31 \cdot 10^{-12}$	$4.51 \cdot 10^{-5}$	$9.30 \cdot 10^{-11}$	$8.90 \cdot 10^{-13}$		

(a) Median search times (in seconds)

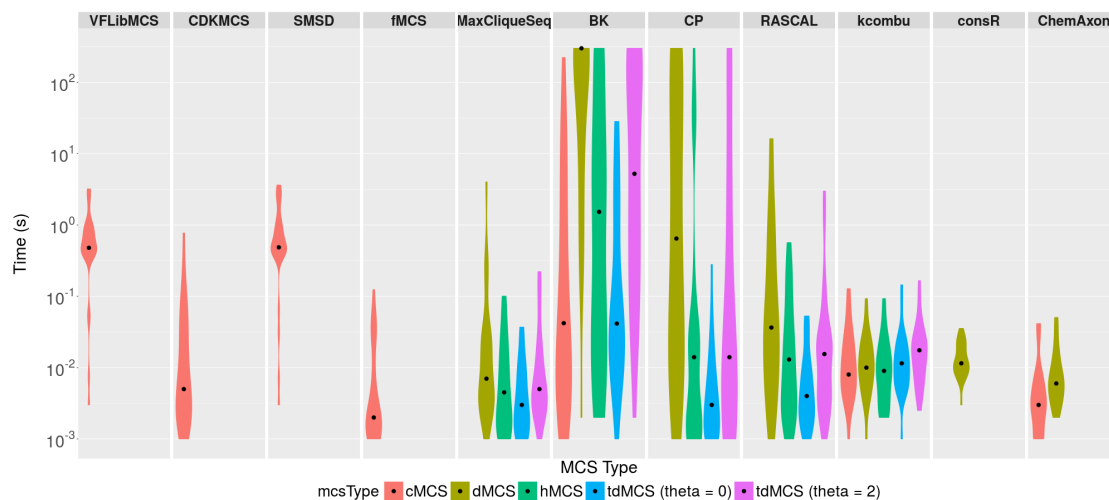
row ID	BK	CP	kcombu	MaxCliqueSeq	RASCAL	W	p
dMCS	14.0	17.0	15.0	17.0	17.0	0.46	$2.46 \cdot 10^{-9}$
hMCS	15.0	17.0	16.0	17.0	17.0	0.29	$9.72 \cdot 10^{-6}$
tdMCS ($\theta = 0$)	16.0	16.0	15.0	16.0	16.0	0.09	$6.11 \cdot 10^{-2}$
tdMCS ($\theta = 2$)	15.0	17.0	15.0	17.0	17.0	0.22	$2.45 \cdot 10^{-4}$
W	0.02	0.21	0.03	0.42	0.42		
p	$6.64 \cdot 10^{-1}$	$1.29 \cdot 10^{-3}$	$5.52 \cdot 10^{-1}$	$6.40 \cdot 10^{-7}$	$7.21 \cdot 10^{-7}$		

(b) Median MCS sizes

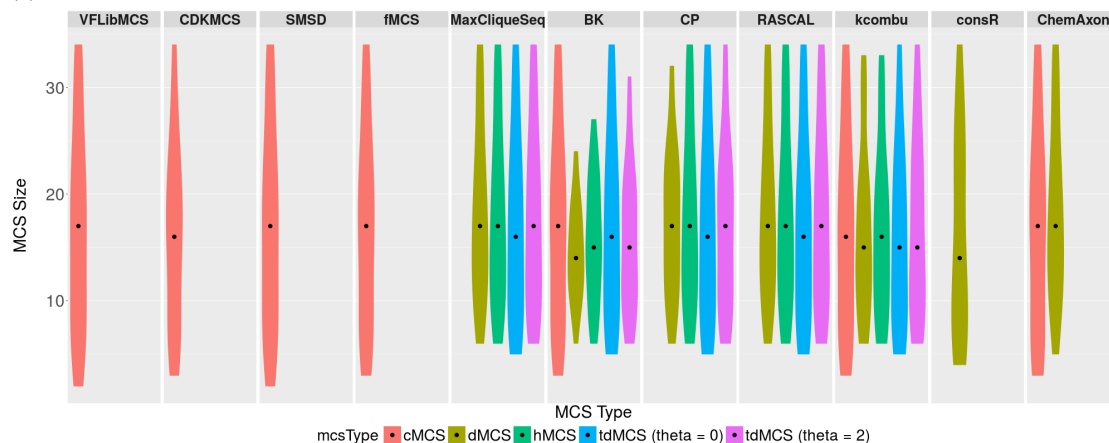
Table 6.9

Comparisons between MCS types (or modular product modifications) for the Franco compounds, formatted in the same way as in Table K.2.

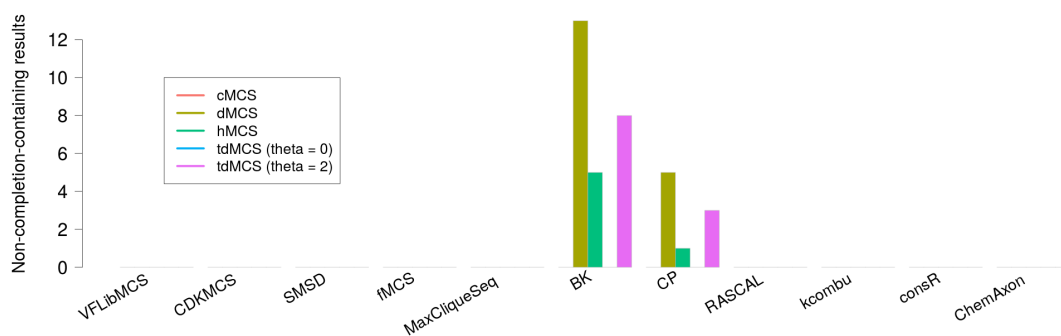
time performance for tdMCS ($\theta = 0$), implying that CP is better suited to more trivial MCS solutions (where the modular product is simplified) than RASCAL and MaxCliqueSeq. This is in contrast to the S, N and M dataset where for tdMCS ($\theta = 0$), CP sometimes outperformed the other algorithms, but in general was not the fastest. Other than these points however, the conclusions from this dataset compared to the S, M and N compounds are otherwise the same.



(a) Time taken



(b) MCS size



(c) Number of non completions for a method.

Figure 6.3. Charts showing performance information of each algorithm (including the same algorithm ran on different MCS types) for the Franco compound pairs.

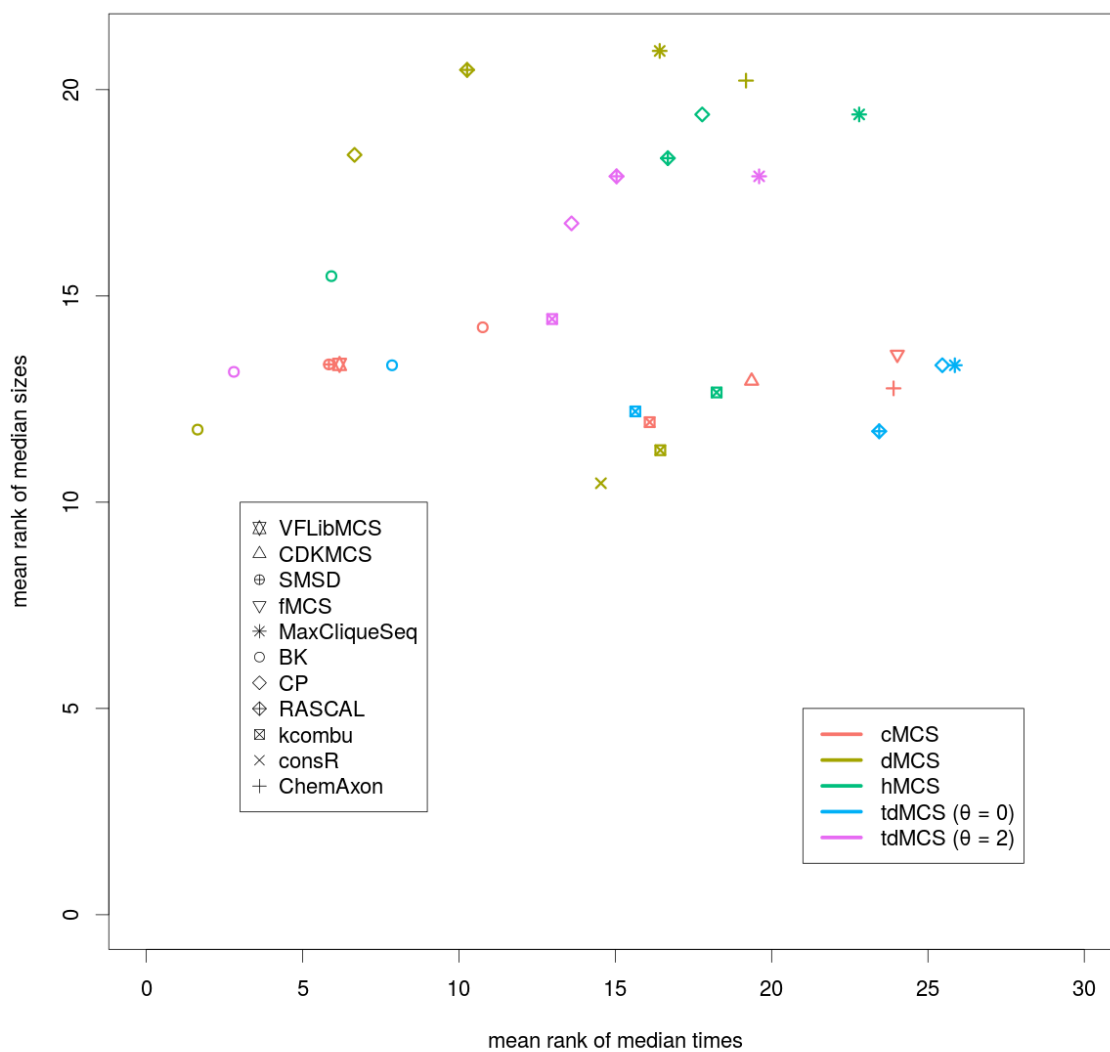


Figure 6.4. Plot showing algorithm-MCS types comparing mean size ranks and time, over each of the Franco pairs. A larger rank implies better performance (so lower time and larger MCS size).

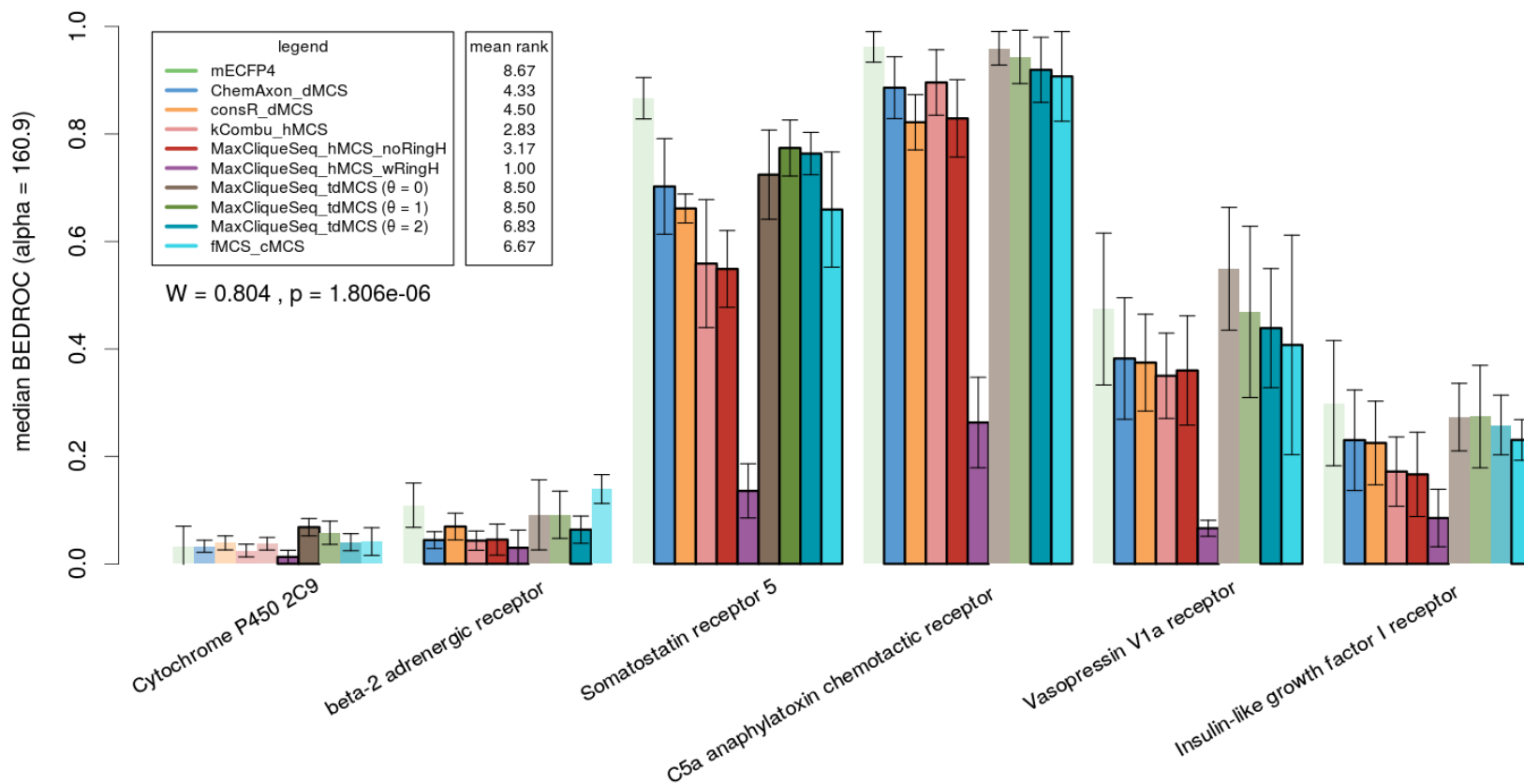


Figure 6.5. Median BEDROC scores for each MCS method (and fingerprints), for each activity class on the virtual screening dataset. Error bars represent 1 median absolute deviation, from the median. Highlighted bars with bold borders indicate that the method is significantly different from that of FP ($p \leq 0.05$), as determined by a paired 2-tailed Wilcoxon signed-rank test. Kendall's W statistics (W and p) are included with the mean ranks (for each method across the classes, where a higher rank indicates better performance). The W test used the results from each activity class as the judges, and the algorithms as the objects.

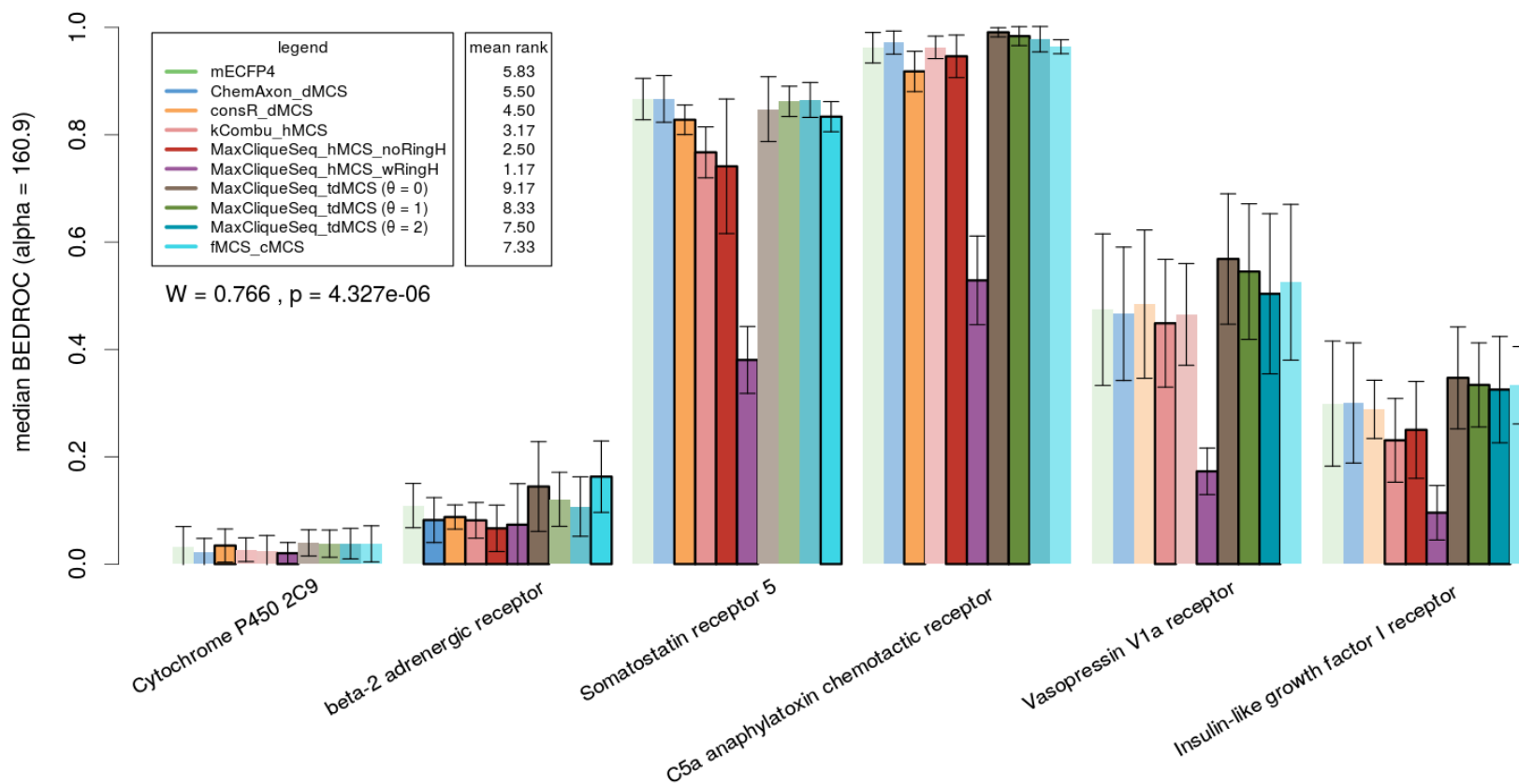


Figure 6.6. Median BEDROC scores for each MCS method (and fingerprints) on the virtual screening dataset, where each method is fused with mECFP4 fingerprints (apart from the fingerprints themselves).

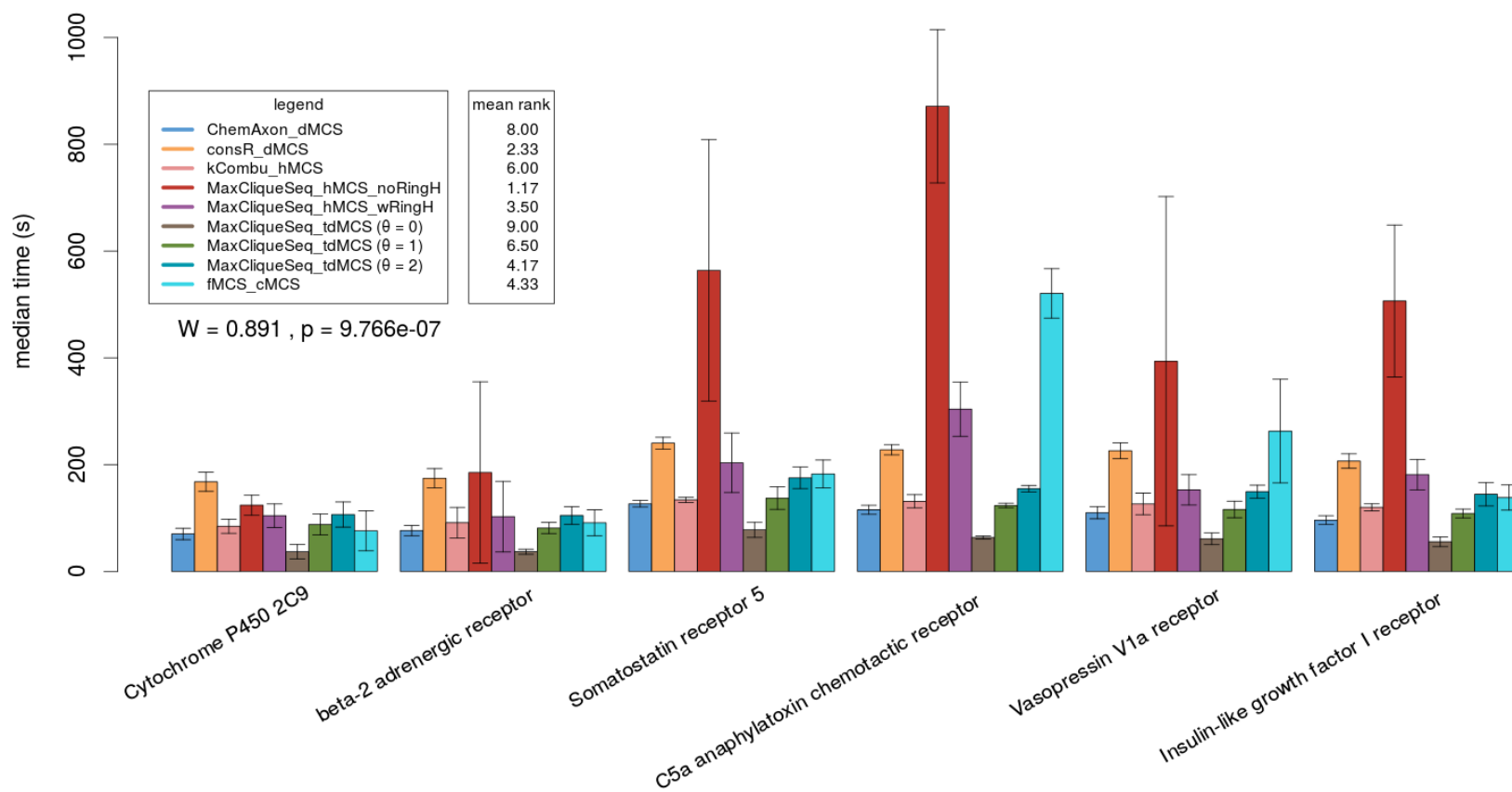


Figure 6.7. Median times scores for each MCS method, on the virtual screening dataset. For the mean ranks shown, a higher rank indicates a shorter time.

6.3.3 Virtual Screening

Median results for BEDROC, BEDROC for the fingerprint-fused algorithms, and times are shown in Figures 6.5, 6.6 and 6.7 respectively. The same figures also show the mean ranks associated with the results. Reflecting back on the methodology section, in addition to the best-performing algorithms from the comparisons of algorithms on the two sets of compound pairs (particular consideration to Figures 6.2 and 6.4), our algorithm choices were as follows for virtual screening:

- MaxCliqueSeq as the exact clique detection method.
- fMCS as the exact cMCS method
- The three approximate dMCS methods.

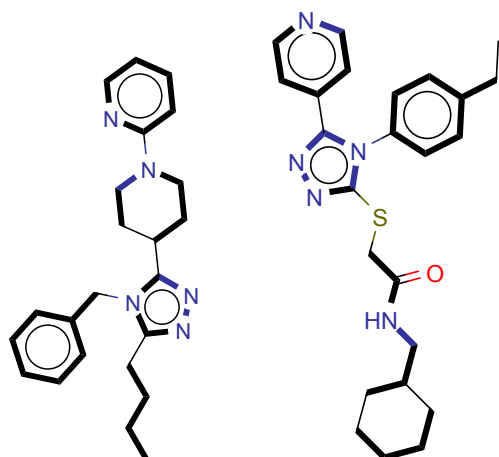
kcombu was executed using hMCS, as from the results of the other datasets, hMCS was generally larger than dMCS for kcombu. As noted in the methodology section, the weak and strong ring heuristics influence the MCS, and hence hMCS has been tested with and without them.

Some interesting conclusions can be drawn from these results. Of immediate relevance is that fingerprints represent the best technique in terms of recall, though tdMCS runs (θ values of 0 and 1) were competitive with fingerprint BEDROC scores (judging from the similar ranks between some MCS algorithms and fingerprints, as well as the high number of insignificant differences from fingerprints shown in Figure 6.5). One should also note that the θ values of 0 and 1 were comparable to, if not quicker in time to the other MCS algorithms. The performance of tdMCS over conventional dMCS has been previously observed by Kawabata (2011), where it was found to give better correspondences than the dMCS, to alignments derived from receptor-ligand complexes. Klinger and Austin (2005) also showed that applying topological constraints in MCS searching yielded better active compound recall than without said constraints. The hMCS, from these results, is evidently unsuitable for virtual screening, perhaps due to the presence of unusual ring systems being compared or a neglect of flexible chains. As shown from the BEDROC results, disabling the weak and strong ring heuristics consistently improved active compound retrieval at a cost in time performance when hMCS was used. The cMCS method was outperformed by tdMCS, though interestingly outranked the dMCS methods tested here. This makes the cMCS a potentially viable option in virtual screening when imposing topological constraints is unfeasible.

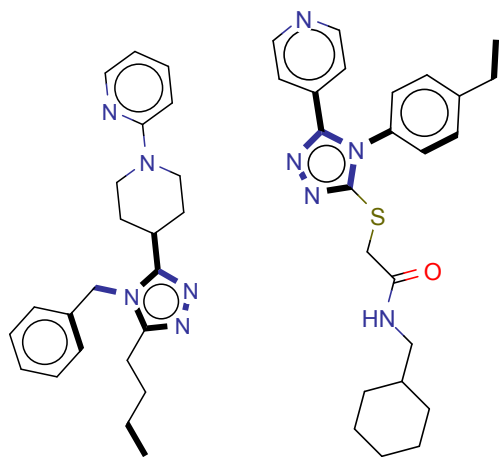
Similarity fusion (SUM rule, using the rank of the best-ranked compound from each of the fingerprint and MCS method) was also performed, the BEDROC results being shown in Figure 6.6. All the tdMCS-fingerprint results, in addition to cMCS-fingerprints, outranked fingerprints alone. tdMCS with $\theta = 0$ however is the result that most consistently outranked fingerprints, where in four of 6 cases, it outperformed them (the other 2 cases had no significant difference from fingerprints). Combined with having the highest rank of all the fusion techniques (as well as being the fastest MCS technique in this study), tdMCS with $\theta = 0$ appears to be the best MCS method to fuse with fingerprints, in order to achieve a slight, but significant and consistent improvement in active compound recall. cMCS-fingerprints fusion, whilst often outranking fingerprints alone, rarely did so on a statistically significant level. Still, when imposing topological constraints is unviable, this cMCS-fingerprint fusion method looks like a potential alternative for improved virtual screening recall.

We note that the performance of the inexact algorithms was often better than exact hMCS calculation (with MaxCliqueSeq), despite the inexact solutions often having smaller sizes (implied from results of the S, N and M pairs, as well as with the Franco pairs). This implies that contrary to intuition - *a larger MCS does not necessarily convey a more sensible similarity* when translating to biological activity. One notable example is comparing ChemAxon's method with consR, where despite ChemAxon's method generally retrieving larger common substructures, the difference in BEDROC scores are small (noting their similar mean ranks). An extreme example using the same algorithms is shown in Figure 6.8, noting that ChemAxon's MCS method yielded a very large MCS whereas consR yielded a very small one (which considering an active-inactive comparison, is more realistic). The same can be seen with a tdMCS, which is also considerably smaller than ChemAxon's result. This is an important observation - if one considers using the MCS for virtual screening, then chemical feasibility should be taken into account. Although the exact dMCS was not calculated in this study, the data here suggest that it is not worth finding it when performing virtual screening, and that inexact algorithms are viable alternatives.

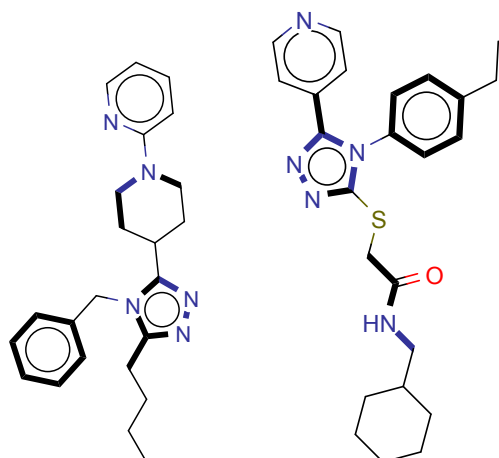
It is promising to note that several of the methods tested here were fast enough to warrant screening of large compound datasets (approaching one million compounds). In effect, for each activity class there were 50475 comparisons performed, being that there are 10095 compounds (10000 decoys and 100 actives, without the 5 reference actives) tested against each reference compound. This implies that ChemAxon's method, using the C5a class as an example, needed no more than 2.29 milliseconds per comparison (115 687 milliseconds divided by 50745 comparisons). Of course, this has been performed using 8 cores in parallel and thus will vary across different computers, but otherwise we



(a) ChemAxon_MCS



(b) consR



(c) MaxCliqueSeq (tdMCS, $\theta = 1$)

Figure 6.8. MCES of a Vasopressin V1a receptor active compound (CHEMBL1837022, left) against an inactive molecule (ZINC09507552), using different MCS algorithms.

see this as a positive result.

6.4 Conclusions

We have identified the fastest-performing MCS algorithms in this work for dMCS (and hMCS), tdMCS and cMCS. MaxCliqueSeq was the fastest exact algorithm for disconnected MCS types whereas fMCS, as tested here, was the quickest exact cMCS algorithm (ChemAxon_MCS on cMCS mode, was the fastest approximate cMCS algorithm). Imposing progressively stricter topological distance constraints improved the speed of algorithm convergence considerably, at a cost in MCS size. As seen from virtual screening results however - the size is not necessarily important, seeing that topological distance limits of 0 and 1 generally yielded results superior to the dMCS approximations used here. Notably, the tdMCS at these two distance limits were competitive with the ChemAxon dMCS algorithm for time performance. ChemAxon's algorithm however remains the method of choice for calculating (the nearest approximation to) the dMCS. hMCS, whilst certainly improving MCS calculation speeds, possesses some flaws that affect the size of the MCS retrieved, and thus requires refinements for determining sensible chemical similarity. Immediate areas to look at are treatment of rigid and flexible systems by the deletion heuristics, as well as for complex ring systems. It is advised that the weak and strong heuristics originally developed by Raymond et al. (2002a), are not used for virtual screening.

As mentioned in Chapter 4, fusion of ChemAxon's dMCS with extended connectivity fingerprints improved active compound recall. In this work, we have shown that fusing the tdMCS with extended connectivity fingerprints actually outperformed extended connectivity fingerprints alone; something which dMCS-fingerprint fusion did not do. This was more noticeable as the topological constraints imposed were progressively stricter. Combined with the fast speed of calculating the tdMCS, this represents a new and superior virtual screening technique.

This work also sheds light on the question of *what a useful MCS actually is*, seeing that inexact algorithms actually were not much worse than exact methods in active compound retrieval. It is advisable for future MCS methods, if tailored for virtual screening, to focus more on sensible chemical (and biological) similarity, than MCS size. Some relevant improvements are suggested in Chapter 7.

In summary, we hope that the work here presents the MCS as a viable complement to fingerprint similarity searching and virtual screening.

Chapter 7

Conclusions

7.1 Summary

This work has explored the use of the MCS in similarity searching, comparing group fusion with various forms of hyperstructure searches. The three objectives outlined in the introduction will each be answered.

7.1.1 MCS-based Hyperstructure Construction

Following the methodology of R. D. Brown, Downs, et al. (1994), a method was devised for deterministically calculating highly-compressed hyperstructures, as presented in chapter 4. It would have been desirable to test alternative orderings of molecules using different MCS types, though the main focus of the work was on similarity searching rather than hyperstructure construction. As shown in chapter 5, a method for constructing frequency-weighted hyperstructures was also devised, as well as the encoding of bond origin information (for ghost substructure elucidation), and the encoding of topology type for a bond (ring or chain type, depending on the properties of the bonds overlapped to form the hyperstructure bond).

7.1.2 Evaluation of MCS Algorithms and Hyperstructures in Similarity Searching

The MCS algorithm used for hyperstructure studies in chapters 4 and 5 was a high performance inexact algorithm developed by ChemAxon. Using this MCS algorithm, it was significantly faster to perform searches with a hyperstructure as opposed to MCS group fusion, using 10 reference molecules. It remains to be seen whether this is also the case with other MCS algorithms, noting

that the ChemAxon algorithm terminates after 1000 iterations - suggesting that larger molecule pairs are searched less thoroughly than smaller ones (Englert & Kovács, 2015).

Several MCS algorithms and MCS types were compared for time and size performance in chapter 6. The fastest exact clique detection algorithm was that from Depolli et al. (2013), particularly when used with a set of modular product simplification heuristics - attaining satisfactory performances on an internally collated set of challenging molecules for MCS algorithms, though in some cases with a decrease in MCS size compared to the dMCS solutions identified. Applying strict topological distance constraints (tdMCS) significantly reduced computation time of the MCS in clique detection algorithms, though understandably would often result in smaller solutions than the dMCS. fMCS was found to be the fastest exact cMCS algorithm, surpassing all of the cMCS algorithms currently found in CDK.

7.1.3 Comparing Virtual Screening Ability of MCS techniques with Fingerprints

The work in chapter 4 showed that our hyperstructure methodology was inferior in virtual screening to conventional fingerprint group fusion, and even to ChemAxon dMCS group fusion. All three approaches were complementary however, in that each retrieved non-overlapping sets of active compounds. Similarity fusion of the fingerprint and MCS techniques generally resulted in an improvement in active compound recall - a finding also noted by Raymond and Willett (2002a). Asymmetric similarity biased towards substructure similarity was found to be significantly preferable to symmetric similarity when concerning hyperstructure similarity - likely due to the hyperstructure being bigger than the vast majority of database structures aligned to it. In chapter 5, a logarithmic frequency weighting scheme was identified as being somewhat superior to binary weighting for active compound recall, though this still did not bring the technique on a par with fingerprint performance. Our results contrasted with the work of Klinger and Austin (2006), who showed that a weighted hyperstructure performed no worse than an unweighted one, and that it also yielded comparable results to MCS group fusion. However, their study also used the tdMCS ($\theta = 0$), which as shown in chapter 6 has preferable virtual screening properties to the dMCS. Presumably the application of the tdMCS yields less unhelpful ghost substructures which allow the retrieval of irrelevant compounds. In addition, their study only used a linear weighting scheme, instead of comparing five weighting schemes as performed here. Simple frequency weighting we note (referred to as W2 in chapter 5) performed worse than binary weighting.

Chapter 6 demonstrated that the only MCS technique competitive with fingerprint group fusion in

virtual screening, was the tdMCS, with the cMCS and dMCS having significantly worse recall statistics. In addition, an exact algorithm was found not to be necessary for virtual screening, since inexact algorithms were found in some cases to outperform exact algorithms. Fusion of tdMCS, cMCS and dMCS results with fingerprints however improved recall, noting that tdMCS-fingerprint fusion was significantly (albeit slightly) better than fingerprints alone, particularly when the topological distance constraint was made stricter.

7.2 Limitations

Some questions arose as to why hyperstructures were outperformed by both fingerprint and MCS group fusion. Chapter 4 highlighted that hyperstructures fail to retrieve analogues of compounds, which the MCS and fingerprints are far better suited to retrieving. However, why is it that even the MCS was outperformed by fingerprints in similarity searching, when the MCS seems like a more intuitive similarity method? Again, this highlights the fact that structural similarity doesn't perfectly correlate with biological "similarity." Using an example where a Vasopressin V1a receptor active compound has been mapped to an inactive compound (Figure 7.1), one will see that there is a large overlap despite the difference in activity. Several non-specific features exist in druglike molecules which may otherwise be irrelevant to specific activity. For example, three benzene rings are in the MCS, which often are responsible for hydrophobic (and otherwise non-specific) interactions in binding sites. An extended connectivity fingerprint would only encode a benzene ring (or any substructure) once, reducing the effect of redundant features. The same problem exists with hyperstructures, though as hyperstructures are bigger than their input molecules, there is a greater chance for non-specific features to match another molecule. This emphasises the importance of avoiding simply using the dMCES to represent similarity. Even the cMCES may be a better option, simply because it is smaller (and noting that it generally outperformed ChemAxon's dMCES method in chapter 6).

It should be noted that although fingerprints and graph theoretic techniques (like the MCS) are generally adequate similarity searching techniques, they ultimately do not capture all the information required for determining the activity of a compound when it concerns the binding of a receptor. Both techniques rely on topological, as opposed to geometrical information to estimate activity. Physico-chemical properties such as the enthalpy of binding, the entropy of the system and solvation effects are ultimately ignored by both techniques, yet they are often crucial for accurate activity determination. Ultimately this is why such techniques are better geared towards the finding of analogues, as opposed to scaffold hopping (as shown in chapter 4). This does not mean the techniques should be avoided in

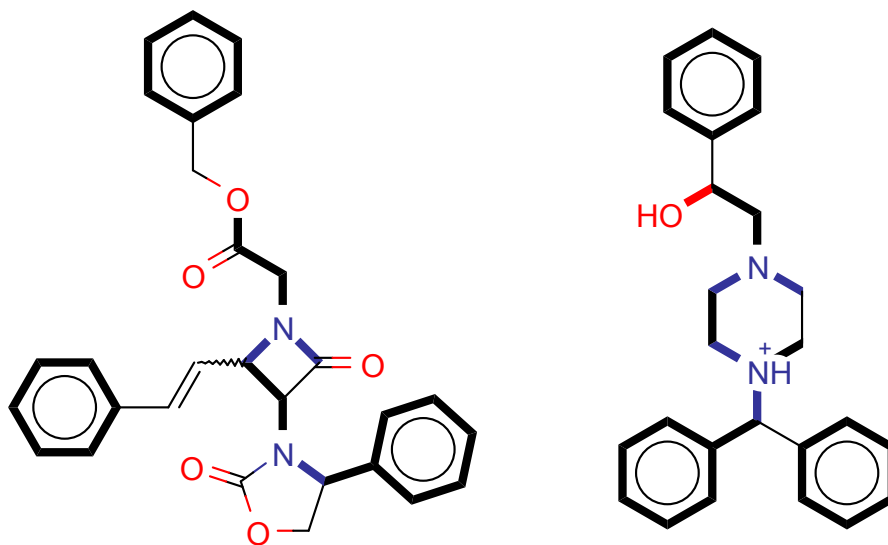


Figure 7.1. MCS of an inactive molecule (ZINC19834655, left) against a Vasopressin V1a receptor active compound (CHEMBL242516).

similarity searching however, for both techniques are often faster than 3D techniques. As an example, the MCS can be used to aid 3D coordinate generation. The work of Kawabata and Nakamura (2014) presents a relevant use of the MCIS in 3D coordinate generation, where torsion angles were copied from a bioactive conformation, to a query molecule. The MCIS was used as a basis for determining which torsion angles to copy to which atoms on the query. By analogy, the MCS could be used to copy coordinates for other 3D techniques, such as docking - where the geometry of a common scaffold could be inferred via the MCS.

7.3 Recommendations for Future Investigations

Although fingerprints are generally a powerful 2D technique for virtual screening, there have been several cases where both fingerprint and MCS techniques have yielded poor recalls in this thesis, particularly when concerning heterogenous activity classes. This stems from the lack of receptor information that is necessary to accurately predict bioactivity, for as discussed in chapter 3, structural similarity does not always accurately predict biological similarity. A number of adjustments to the hyperstructure and MCS techniques have been proposed below, which we have not discussed in depth elsewhere in the thesis.

7.3.1 Weighted MCS

The thesis of John Raymond (Raymond, 2002) suggested the idea of allowing bonds of differing atom types and bond types to match. The resulting bond pairs from the MCS would then be assigned weights from a bond substitution table, where different bonds would be assigned weights based on their chemical similarity. Weights can be assigned using a pre-calculated substitution table, as done by Berglund and Head (2010) and Teixeira and Falcao (2013). In the case of finding active molecules however, it would be more useful to empirically determine weights for matching bond types, based on their relative occurrences in active and inactive compound alignments. This of course would require a set of reference alignments to calculate such weights. It is an idea that is similar to that used by the PAM and BLOSUM amino acid substitution matrices in protein sequence alignment (Henikoff & Henikoff, 1992). The BLOSUM matrices are of particular interest as they are derived from *local* alignments of conserved regions found in proteins of the same family. Translating to a chemoinformatics perspective - this would involve deriving weights from MCS alignments of local regions in active compounds (for instance, the scaffolds of the compounds). Said weights could then be used in a similarity coefficient, such as the Tanimoto coefficient, to yield a modified similarity. Weights calculated in this fashion should act not only to upweight important features found in active molecules, but also to downweight common features found in both actives and inactives (such as benzene rings, which are almost ubiquitous in drug-like molecules). On a related note, one does not necessarily need to represent atoms via atomic number. A suitable alternative would be functional (also known as pharmacophore) classes, as defined by Rogers and Hahn (2010).

7.3.2 Improvement of Clique Detection Algorithm

Although the clique detection algorithm of Depolli et al. (2013) was shown to be fast in this thesis, it could still be made faster for the computation of certain difficult cases, particularly when one wishes to find the MCS for large compounds. The algorithm in question notably uses no chemical knowledge to improve the speed of the algorithm.

7.3.2.1 Graph colouring. The first adjustment to try is to use the partitioning method in RASCAL, instead of the greedy colouring algorithm in the original algorithm, which from internal tests we found to use consistently less “colours” than the greedy algorithm. In addition, the label-based upper bound of RASCAL derived from the modular product would yield a more accurate upper bound than the colouring algorithm. This is in light of the fact that Raymond et al. (2002b) noted that a

greedy colouring algorithm became progressively inaccurate as an upper bounding procedure outside the range of 125 to 175 nodes in a given graph.

7.3.2.2 Initial Lower Bound. The original algorithm derived an initial lower bound upon finishing the vertex ordering procedure - where the identification of an initial clique would result. Whilst this is an efficient process, it may be worth using the result from an inexact algorithm as an initial lower bound instead. A suitable candidate would be kcombu, which often would obtain the MCS anyway as observed in chapter 6. The sharper lower bound will otherwise improve the speed of the exact algorithm, despite the slight overhead from running the inexact algorithm first. Alternatively, if one does not wish to use chemical information, then a maximum clique derived from the minimum vertex cover of a complement graph could be used as an alternative lower bound (as explained in chapter 2).

7.3.2.3 Ordering of Search Tree. Modular product nodes were originally ordered in descending order of degree and ex-degree. Englert and Kovács (2015) noted that employing a form of ECFP fingerprint-based similarity to select nodes in clique detection improved the search speed and result size. We suggest to use fingerprint similarity to order the nodes in the modular product, even when matching atoms and bonds of different labels. One way this could be performed is by taking the neighbourhoods around the atom/bond pair, and calculating the similarity of each neighbourhood. Nodes with high similarities would be ordered first. We would recommend comparing the neighbourhoods using path-based fingerprints instead of radial fingerprints to ignore the topology matching effect (i.e. we want rings and chains to match). Tanimoto similarity should suffice for neighbourhood comparisons, but alternative coefficients could be used instead. Likewise, the bias towards c-edges suggested by the same authors could be exploited, though we suggest that c-edge number is calculated to select nodes during clique expansion, rather than for ordering the nodes in the modular product.

7.3.3 Hyperstructure Improvements

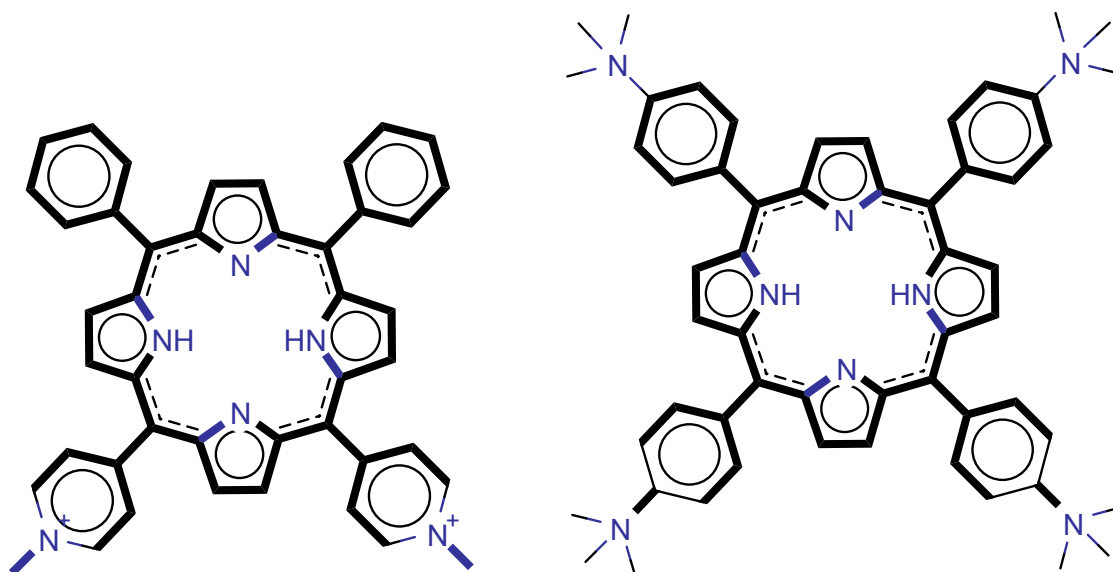
The thesis of Nathan Brown (N. Brown, 2002) had a number of interesting suggestions on future work for the hyperstructure concept. This included the application of hyperstructures via reduced graphs (specifically, the creation of reduced hyperstructures via the overlap of reduced graphs).

7.3.3.1 Effects of other MCS types. We already know that applying topological constraints improves the search performance of hyperstructures, though it would be interesting to vary the strictness of the tdMCS applied in both hyperstructure construction, and application (Klinger & Austin, 2006).

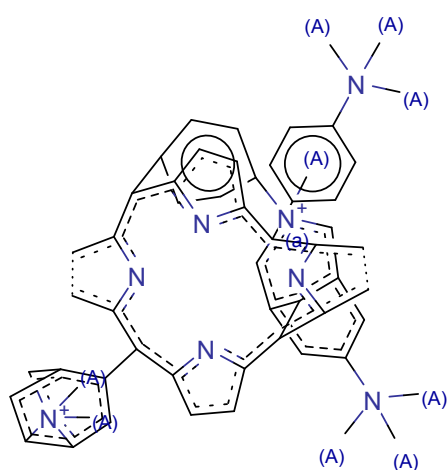
This could be translated not only to similarity searching, but also to QSAR modelling based on MCS alignment (Palyulin et al., 2000) - particularly amongst molecules of different chemical series, which to our knowledge has not been attempted with hyperstructures yet. An example of the benefits with using the tdMCS is demonstrated in Figure 7.2. The hyperstructure resulting from the tdMCS is clearly more chemically feasible than the dMCS one. Granted that chemical feasibility is not necessarily a concern in hyperstructures, one should appreciate the fact that the tdMCS yields a better-quality alignment.

When using the tdMCS to map molecules to hyperstructures, one should be aware that simply using the topological distances derived from the hyperstructure can cause complications; the shortest path from two features obtained in a reference molecule compared to the shortest path of its matching features in its hyperstructure, may be different lengths. Consider the example in Figure 7.3. The bonds marked as *, † and ‡ between each of the 3 molecules correspond to each other, respectively (i.e. The bond * in G_1 matches * in G_2 , which in turn matches * in the hyperstructure G_H). In G_1 , the topological distance (using the shortest path) between * and † is 9. In G_H as well as G_2 however, it is 7. If using a strict tdMCS (for instance, where $\theta = 0$), then G_1 will not be treated as subgraph isomorphic to G_H due to the path distance differences. Therefore, when constructing a hyperstructure, path distances from the original graph(s) between two bonds should be kept in addition to the distance between them in the hyperstructure. Thus, for the bonds * and † the distances stored should be {7, 9} when calculating a topologically constrained MCS. Likewise for the pairs * and ‡ the distance in G_1 is 7, whereas in G_H it is 5, thus in the hyperstructure the distance stored should be {5, 7}.

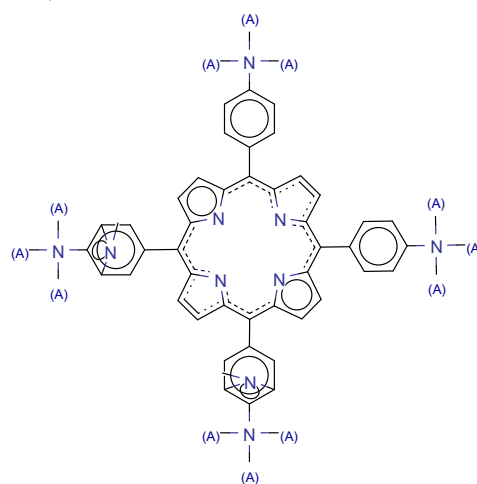
No form of penalisation has been imposed to take into account the number of fragments present in the dMCES, yet chemically it is more realistic to say that two molecules are more similar when they share a large fragment in common, as opposed to several disconnected fragments (Figure 7.4). When used by Raymond and Willett (2002a), fragment penalisation slightly but nevertheless consistently improved active compound. This idea has so far not been implemented due to the inexact nature of the MCS algorithm used in this work. However, the exact MCS algorithm proposed by Hariharan et al. (2011) which seeks to find the MCS with the lowest number of fragments, may be of significant aid in fragment penalisation. In relation to this idea, it may be worth just using the cMCES instead of the dMCES currently used to compute similarity, avoiding the penalisation problem by using only the largest fragment in common. This however is expected to yield worse results as the potential for scaffold hopping could be hampered.



(a) Compound pair (S6 in chapter 6) from (Libby et al., 1995).



(b) Hyperstructure constructed using a dMCES (ChemAxon's dMCES method)



(c) Hyperstructure constructed using a tdMCES where $\theta = 1$ (calculated with MaxCliqueSeq)

Figure 7.2. Two hyperstructures from the S6 pair in chapter 6, where the MCS used to produce the hyperstructure originates from the different methods, yet both MCSs have the same number of bonds.

7.3.3.2 Ghost Substructures. Chapter 5 showed that the proportion of ghost substructures found in the MCS between hyperstructures and database molecules is different between active and inactive compounds. It would be worth exploring this further to attempt to observe more effects of ghost substructures, as well as seeing whether different MCS types affect the proportions of ghost substructures identified (for both the purposes of hyperstructure construction, and similarity searching). In addition,

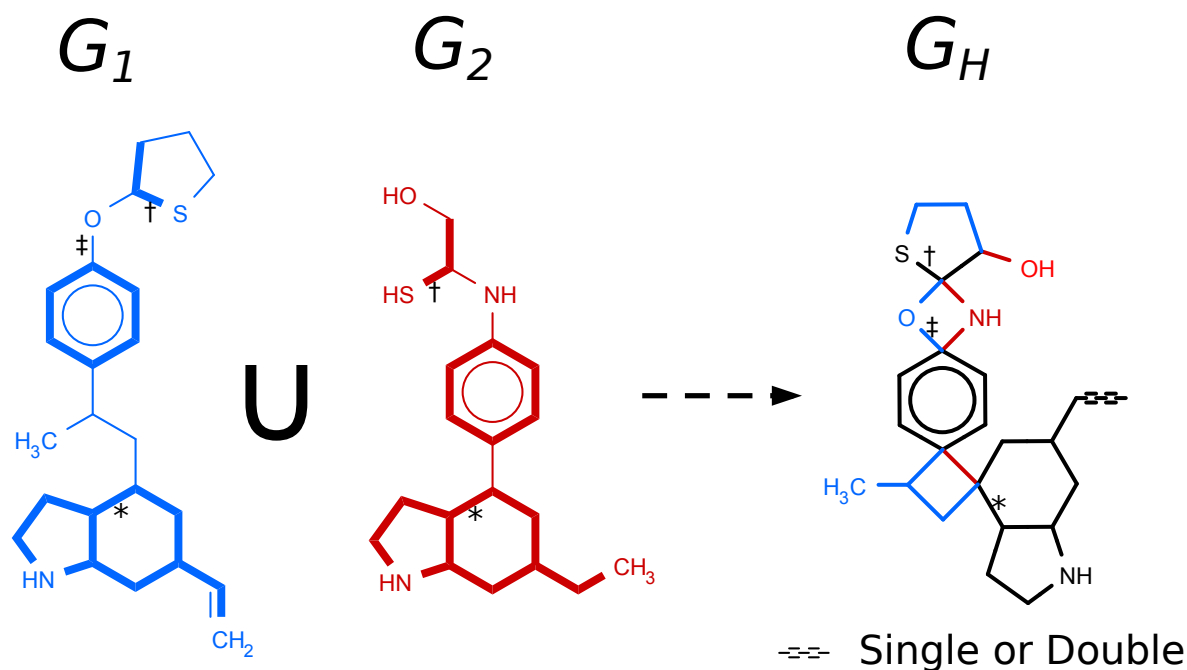


Figure 7.3. Example adapted from the hyperstructure construction figure in chapter 4, to illustrate topological distance complications in hyperstructures.

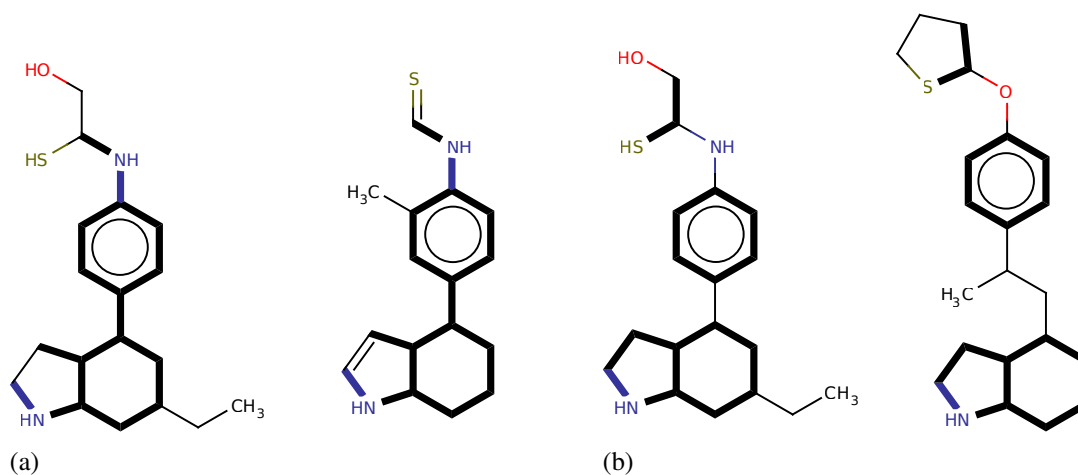


Figure 7.4. Two pairs of molecules where the bonds in the MCES are shown in bold, both pairs having 18 bonds in the MCES. The pair in (7.4a) are chemically more similar in this example than the other pair, for the MCES consists of one large connected fragment, whereas the other pair consists of 3 connected fragments.

mapping of database molecule edges to ghost substructures for example, could be penalised. Ghost substructure occurrences could also be reduced by allowing atoms to map to other atoms of different labels (and likewise for bonds).

7.3.3.3 Labelling. Although this also concerns MCS alignments, it would be a wise idea to experiment with alternative graph labelling and weighting. The MCSs calculated here used atomic

number and bond type for labelling and weighting respectively, yet one could instead use functional or pharmacophoric atom types (Rogers & Hahn, 2010), and bond lengths instead of bond order. Hyperstructures of course could be constructed from such alternatively-labelled molecules, and would be expected to perform differently in similarity searching compared to the hyperstructures currently used in this work. Likewise, we have only considered mapping bonds of different weights (or orders) to each other, yet we have not considered the mapping of different atom labels.

7.3.3.4 Fingerprints. Hyperstructures to date have not been represented in any form using fingerprints. It may be preferable to actually perform similarity searching by abstracting the hyperstructure into path-based fingerprints. Radial fingerprints would represent a bad choice of fingerprint as rings in hyperstructures have the potential to reach infinitely large sizes. Due to the large degrees that some nodes exhibit in hyperstructures (as well as their general complexity), fingerprints produced would be much denser than with conventional molecules. This point could potentially give rise to erroneous results due to a greater potential for yielding ambiguous bits, in addition to giving a slower search time.

7.4 Final Remarks

Several variations of the MCS exist in chemoinformatics, tailored towards different uses. Although the elucidation of the MCS is an NP-complete problem, much recent progress has been made in algorithm development to quicken the finding of an MCS, which in turn has made it more viable for similarity searching. Although the hyperstructure (via dMCS mapping) has proved to be relatively unsuccessful in similarity-based virtual screening in this thesis, it remains to be seen as to how alternative MCS types will affect its performance. In addition, the tdMCS and cMCS have been shown here to be superior to the (exact) dMCS in active compound recall, and fusion of the tdMCS with fingerprints outperforms fingerprints alone.

It is hoped that the work of this thesis will guide others who are exploring the hyperstructure concept, both in and outside of similarity searching. It is also hoped that this will highlight which MCS type and algorithm to use for similarity searches, in addition to non-similarity based methods (namely reaction mapping and molecular alignment).

References

- Adamson, G. W., Cowell, J., Lynch, M. F., McLure, A. H. W., Town, W. G., & Yapp, A. M. (1973). Strategic Considerations in the Design of a Screening System for Substructure Searches of Chemical Structure Files. *Journal of Chemical Documentation*, 13(3), 153–157. Retrieved 2015-08-27, from <http://dx.doi.org/10.1021/c160050a013> doi: 10.1021/c160050a013
- Adler, I., Dorn, F., Fomin, F. V., Sau, I., & Thilikos, D. M. (2010). Fast Minor Testing in Planar Graphs. In M. d. Berg & U. Meyer (Eds.), *Algorithms – ESA 2010* (pp. 97–109). Springer Berlin Heidelberg. Retrieved 2015-03-23, from http://link.springer.com/chapter/10.1007/978-3-642-15775-2_9
- Aldridge, K., George, I. D., Cole, K. K., Austin, J. R., Takahashi, T. N., Duan, Y., & Miles, J. H. (2011). Facial Phenotypes in Subgroups of Prepubertal Boys with Autism Spectrum Disorders Are Correlated with Clinical Phenotypes. *Molecular Autism*, 2(1), 15. doi: 10.1186/2040-2392-2-15
- Al Khalifa, A., Haranczyk, M., & Holliday, J. (2009). Comparison of Nonbinary Similarity Coefficients for Similarity Searching, Clustering and Compound Selection. *Journal of Chemical Information and Modeling*, 49(5), 1193–1201. Retrieved 2013-09-11, from <http://dx.doi.org/10.1021/ci8004644> doi: 10.1021/ci8004644
- Appel, K., & Haken, W. (1977). Every Planar Map Is Four Colorable. Part I: Discharging. *Illinois Journal of Mathematics*, 21(3), 429–490. Retrieved 2012-11-23, from <http://projecteuclid.org/euclid.ijm/1256049011>
- Arif, S. M., Hert, J., Holliday, J. D., Malim, N., & Willett, P. (2009). Enhancing the Effectiveness of Fingerprint-Based Virtual Screening: Use of Turbo Similarity Searching and of Fragment Frequencies of Occurrence. In V. Kadiramanathan, G. Sanguinetti, M. Girolami, M. Niranjana, & J. Noirel (Eds.), *Pattern Recognition in Bioinformatics* (pp. 404–414). Springer Berlin Heidelberg. Retrieved 2013-11-14, from http://link.springer.com/chapter/10.1007/978-3-642-04031-3_35
- Arif, S. M., Holliday, J. D., & Willett, P. (2009). Analysis and Use of Fragment-Occurrence Data in Similarity-Based Virtual Screening. *Journal of Computer-Aided Molecular Design*, 23(9), 655–668. Retrieved 2013-08-05, from <http://link.springer.com/article/10.1007/s10822-009-9285-0> doi: 10.1007/s10822-009-9285-0
- Arif, S. M., Holliday, J. D., & Willett, P. (2010). Inverse Frequency Weighting of Fragments for Similarity-

- Based Virtual Screening. *Journal of Chemical Information and Modeling*, 50(8), 1340–1349. Retrieved 2013-08-05, from <http://dx.doi.org/10.1021/ci1001235> doi: 10.1021/ci1001235
- Armitage, J. E., & Lynch, M. F. (1967). Automatic Detection of Structural Similarities Among Chemical Compounds. *Journal of the Chemical Society C: Organic*, 521–528. Retrieved 2013-01-09, from <http://pubs.rsc.org/en/content/articlelanding/1967/j3/j39670000521> doi: 10.1039/J39670000521
- Ash, S., Cline, M. A., Homer, R. W., Hurst, T., & Smith, G. B. (1997). SYBYL Line Notation (SLN): A Versatile Language for Chemical Structure Representation†. *Journal of Chemical Information and Computer Sciences*, 37(1), 71–79. Retrieved 2012-11-06, from <http://dx.doi.org/10.1021/ci960109j> doi: 10.1021/ci960109j
- Ashton, M., Barnard, J., Casset, F., Charlton, M., Downs, G., Gorse, D., ... Willett, P. (2002). Identification of Diverse Database Subsets using Property-Based and Fragment-Based Molecular Descriptions. *Quantitative Structure-Activity Relationships*, 21(6), 598–604. Retrieved 2013-07-19, from <http://onlinelibrary.wiley.com/doi/10.1002/qsar.200290002/abstract> doi: 10.1002/qsar.200290002
- Babel, L. (1994). A Fast Algorithm for the Maximum Weight Clique Problem. *Computing*, 52(1), 31–38. Retrieved 2012-11-26, from <http://www.springerlink.com/index/71721L50606K6501.pdf>
- Barker, E. J., Buttar, D., Cosgrove, D. A., Gardiner, E. J., Kitts, P., Willett, P., & Gillet, V. J. (2006). Scaffold Hopping Using Clique Detection Applied to Reduced Graphs. *Journal of Chemical Information and Modeling*, 46(2), 503–511. Retrieved 2014-09-12, from <http://dx.doi.org/10.1021/ci050347r> doi: 10.1021/ci050347r
- Barrow, H., & Burstall, R. (1976). Subgraph Isomorphism, Matching Relational Structures and Maximal Cliques. *Information Processing Letters*, 4(4), 83–84. Retrieved 2012-11-21, from <http://www.sciencedirect.com/science/article/pii/0020019076900491> doi: 10.1016/0020-0190(76)90049-1
- Bemis, G. W., & Murcko, M. A. (1996). The Properties of Known Drugs. 1. Molecular Frameworks. *Journal of medicinal chemistry*, 39(15), 2887–2893. doi: 10.1021/jm9602928
- Berglund, A. E., & Head, R. D. (2010). PZIM: A Method for Similarity Searching Using Atom Environments and 2d Alignment. *Journal of Chemical Information and Modeling*, 50(10), 1790–1795. Retrieved 2013-10-28, from <http://dx.doi.org/10.1021/ci1002075> doi: 10.1021/ci1002075
- Berthold, M. R., Cebon, N., Dill, F., Gabriel, T. R., Kötter, T., Meinel, T., ... Wiswedel, B. (2008). KNIME: The Konstanz Information Miner. In C. Preisach, P. D. H. Burkhardt, P. D. L. Schmidt-Thieme, & P. D. R. Decker (Eds.), *Data Analysis, Machine Learning and Applications* (pp. 319–326). Springer Berlin Heidelberg. Retrieved 2013-08-16, from http://link.springer.com/chapter/10.1007/978-3-540-78246-9_38
- Berthold, M. R., Cebon, N., Dill, F., Gabriel, T. R., Kötter, T., Meinel, T., ... Wiswedel, B. (2009). Knime

- the Konstanz Information Miner: Version 2.0 and Beyond. *ACM SIGKDD Explorations Newsletter*, 11(1), 26–31. Retrieved 2013-08-16, from <http://doi.acm.org/10.1145/1656274.1656280> doi: 10.1145/1656274.1656280
- BIOVIA Databases | Sourcing Databases: BIOVIA Available Chemicals Directory (ACD) [Commercial]. (n.d.). Retrieved 2015-09-03, from <http://accelrys.com/products/collaborative-science/databases/sourcing-databases/biovia-available-chemicals-directory.html>
- Birchall, K., & Gillet, V. J. (2010). Reduced Graphs and Their Applications in Chemoinformatics. In J. Bajorath (Ed.), *Chemoinformatics and Computational Chemical Biology* (Vol. 672, pp. 197–212). Totowa, NJ: Humana Press. Retrieved 2016-01-30, from http://link.springer.com/10.1007/978-1-60761-839-3_8
- Bolton, E. E., Wang, Y., Thiessen, P. A., & Bryant, S. H. (2008). Chapter 12 PubChem: Integrated Platform of Small Molecules and Biological Activities. In R. A. Wheeler & D. C. Spellmeyer (Eds.), *Annual Reports in Computational Chemistry* (Vol. Volume 4, pp. 217–241). Elsevier. Retrieved 2012-11-07, from <http://www.sciencedirect.com/science/article/pii/S1574140008000121>
- Bomze, I. M., Budinich, M., Pardalos, P. M., & Pelillo, M. (1999). The Maximum Clique Problem. *Handbook of Combinatorial Optimization*, 4(1), 1–74. Retrieved 2015-08-22, from http://link.springer.com/chapter/10.1007/978-1-4757-3023-4_1
- Bone, R., Vacca, J. P., Anderson, P. S., & Holloway, M. K. (1991). X-Ray Crystal Structure of the HIV Protease Complex with L-700,417, an Inhibitor with Pseudo C2 Symmetry. *Journal of the American Chemical Society*, 113, 9382–9384. Retrieved 2015-03-09, from <http://pubs.acs.org/doi/abs/10.1021/ja00024a061>
- Borgwardt, K. M., Ong, C. S., Schonauer, S., Vishwanathan, S. V. N., Smola, A. J., & Kriegel, H.-P. (2005). Protein Function Prediction Via Graph Kernels. *Bioinformatics*, 21(Suppl 1), i47–i56. Retrieved 2012-11-20, from http://bioinformatics.oxfordjournals.org/content/21/suppl_1/i47.short doi: 10.1093/bioinformatics/bti1007
- Boström, J., Hogner, A., & Schmitt, S. (2006). Do Structurally Similar Ligands Bind in a Similar Fashion? *Journal of Medicinal Chemistry*, 49(23), 6716–6725. Retrieved 2013-06-14, from <http://dx.doi.org/10.1021/jm060167o> doi: 10.1021/jm060167o
- Brint, A. T., & Willett, P. (1987). Algorithms for the Identification of Three-Dimensional Maximal Common Substructures. *Journal of Chemical Information and Computer Sciences*, 27(4), 152–158. Retrieved 2015-06-17, from <http://dx.doi.org/10.1021/ci00056a002> doi: 10.1021/ci00056a002
- Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, R., ... Wiener, J. (2000). Graph Structure in the Web. *Computer Networks*, 33(1–6), 309–320. Retrieved 2012-11-01, from <http://www.sciencedirect.com/science/article/pii/S1389128600000839> doi: 10.1016/S1389-1286(00)00083-9
- Bron, C., & Kerbosch, J. (1973). Algorithm 457: Finding all Cliques of an Undirected Graph. *Communications*

- of the ACM*, 16(9), 575–577. Retrieved 2012-11-26, from <http://dl.acm.org/citation.cfm?id=362367>
- Brown, A. C., & Fraser, T. R. (1868). On the Connection between Chemical Constitution and Physiological Action; with special reference to the Physiological Action of the Salts of the Ammonium Bases derived from Strychnia, Brucia, Thebaia, Codeia, Morphia, and Nicotia. *Journal of Anatomy and Physiology*, 2(2), 224–242. Retrieved 2015-08-12, from <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1318606/>
- Brown, N. (2002). *Generation and Application of Activity-Weighted Chemical Hyperstructures*. (PhD Thesis, University of Sheffield, Sheffield). Retrieved from http://books.google.co.uk/books/about/Generation_and_Application_of_Activity_w.html?id=L54CSQAACAAJ&redir_esc=y
- Brown, N., Willett, P., Wilton, D. J., & Lewis, R. A. (2003). Generation and Display of Activity-Weighted Chemical Hyperstructures. *Journal of Chemical Information and Computer Sciences*, 43(1), 288–297. Retrieved 2012-12-11, from <http://dx.doi.org/10.1021/ci0103875> doi: 10.1021/ci0103875
- Brown, R. D., Downs, G. M., Jones, G., & Willett, P. (1994). Hyperstructure Model for Chemical Structure Handling: Techniques for Substructure Searching. *Journal of Chemical Information and Computer Sciences*, 34(1), 47–53. Retrieved 2012-10-26, from <http://dx.doi.org/10.1021/ci00017a006> doi: 10.1021/ci00017a006
- Brown, R. D., Downs, G. M., Willett, P., & Cook, A. P. F. (1992). Hyperstructure Model for Chemical Structure Handling: Generation and Atom-by-Atom Searching of Hyperstructures. *Journal of Chemical Information and Computer Sciences*, 32(5), 522–531. Retrieved 2012-11-08, from <http://dx.doi.org/10.1021/ci00009a020> doi: 10.1021/ci00009a020
- Brown, R. D., Jones, G., Willett, P., & Glen, R. C. (1994). Matching Two-Dimensional Chemical Graphs Using Genetic Algorithms. *Journal of Chemical Information and Computer Sciences*, 34(1), 63–70. Retrieved 2012-10-24, from <http://dx.doi.org/10.1021/ci00017a008> doi: 10.1021/ci00017a008
- Bunke, H. (1997). On a Relation between Graph Edit Distance and Maximum Common Subgraph. *Pattern Recognition Letters*, 18(8), 689–694. Retrieved 2013-10-30, from <http://www.sciencedirect.com/science/article/pii/S0167865597000603> doi: 10.1016/S0167-8655(97)00060-3
- Bunke, H. (1998). Error-tolerant Graph Matching: A Formal Framework and Algorithms. In A. Amin, D. Dori, P. Pudil, & H. Freeman (Eds.), *Advances in Pattern Recognition* (pp. 1–14). Springer Berlin Heidelberg. Retrieved 2013-10-30, from <http://link.springer.com/chapter/10.1007/BFb0033223>
- Bunke, H., Dickinson, P., Kraetzl, M., Neuhaus, M., & Stettler, M. (2008). Matching of Hypergraphs — Algorithms, Applications, and Experiments. In J. Kacprzyk, H. Bunke, A. Kandel, & M. Last (Eds.), *Applied Pattern Recognition* (Vol. 91, pp. 131–154). Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved 2016-01-30, from http://link.springer.com/10.1007/978-3-540-76831-9_6
- Bunke, H., Jiang, X., & Kandel, A. (2000). On the Minimum Common Supergraph of Two Graphs. *Computing*, 65(1), 13–25. Retrieved 2012-10-30, from <http://www.springerlink.com/content/>

- awdthf1fyw2wka49/abstract/ doi: 10.1007/s006070050010
- Cao, B., Yang, Q., Sun, J.-T., & Chen, Z. (2011). Learning Bidirectional Asymmetric Similarity for Collaborative Filtering Via Matrix Factorization. *Data Mining and Knowledge Discovery*, 22(3), 393–418. Retrieved 2013-06-21, from http://dblife.cs.wisc.edu/publication/Learning_bidirectional_asymmetric_similarity_for_collaborative_filtering_via_matrix_factorization doi: 10.1007/s10618-011-0211-4
- Cao, Y., Jiang, T., & Girke, T. (2008). A Maximum Common Substructure-Based Algorithm for Searching and Predicting Drug-Like Compounds. *Bioinformatics*, 24(13), i366–i374. Retrieved 2015-02-28, from <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2718661/> doi: 10.1093/bioinformatics/btn186
- Carhart, R. E., Smith, D. H., & Venkataraghavan, R. (1985). Atom Pairs as Molecular Features in Structure-Activity Studies: Definition and Applications. *Journal of Chemical Information and Computer Sciences*, 25(2), 64–73. Retrieved 2012-11-14, from <http://dx.doi.org/10.1021/ci00046a002> doi: 10.1021/ci00046a002
- Carlson, J. A., Jaffe, A., & Wiles, A. (2006). *The Millennium Prize Problems*. American Mathematical Society. Retrieved 2012-12-11, from <http://books.google.co.uk/books?hl=en&lr=&id=7wJIPJ80RdUC&oi=fnd&pg=PR7&dq=http://www.claymath.org/library/monographs/MPPc.pdf&ots=3FDAsbLnkF&sig=LU9LRBjet-9wUPBRiN9KmeONNz4>
- Carraghan, R., & Pardalos, P. M. (1990). An Exact Algorithm for the Maximum Clique Problem. *Operations Research Letters*, 9(6), 375–382. Retrieved 2012-11-26, from <http://www.sciencedirect.com/science/article/pii/S016763779090057C>
- Cayley, A. (1857). XXVIII. On the Theory of the Analytical Forms Called Trees. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 13(85), 172–176. Retrieved 2012-11-05, from <http://www.tandfonline.com/doi/abs/10.1080/14786445708642275>
- Cazals, F., & Karande, C. (2008). A Note on the Problem of Reporting Maximal Cliques. *Theoretical Computer Science*, 407(1–3), 564–568. Retrieved 2015-01-07, from <http://www.sciencedirect.com/science/article/pii/S0304397508003903> doi: 10.1016/j.tcs.2008.05.010
- Cereto-Massagué, A., Ojeda, M. J., Valls, C., Mulero, M., Garcia-Vallvé, S., & Pujadas, G. (2015). Molecular Fingerprint Similarity Search in Virtual Screening. *Methods*, 71, 58–63. Retrieved 2015-02-23, from <http://www.sciencedirect.com/science/article/pii/S1046202314002631> doi: 10.1016/j.ymeth.2014.08.005
- CERN. (2004). *Colt - Welcome* [Content Site]. Retrieved 2015-03-13, from <http://dst.lbl.gov/ACSSoftware/colt/index.html>
- Certara Enhances SYBYL-X Drug Design and Discovery Software Suite* [Commercial]. (2013). Retrieved 2015-02-06, from <https://www.certara.com/news/certara-enhances-sybyl-x-drug-design-and-discovery-software-suite>

- Chen, B., Mueller, C., & Willett, P. (2010). Combination Rules for Group Fusion in Similarity-Based Virtual Screening. *Molecular Informatics*, 29(6-7), 533–541. Retrieved 2013-09-19, from <http://onlinelibrary.wiley.com/doi/10.1002/minf.201000050/abstract> doi: 10.1002/minf.201000050
- Chen, X., & Brown, F. K. (2007). Asymmetry of Chemical Similarity. *ChemMedChem*, 2(2), 180–182. Retrieved 2013-06-20, from <http://onlinelibrary.wiley.com/doi/10.1002/cmdc.200600161/abstract> doi: 10.1002/cmdc.200600161
- Christie, B. D., Leland, B. A., & Nourse, J. G. (1993). Structure Searching in Chemical Databases by Direct Lookup Methods. *Journal of Chemical Information and Computer Sciences*, 33(4), 545–547. Retrieved 2012-11-06, from <http://dx.doi.org/10.1021/ci00014a004> doi: 10.1021/ci00014a004
- Clark, M., Cramer, R. D., & Van Opdenbosch, N. (1989). Validation of the General Purpose Tripos 5.2 Force Field. *Journal of Computational Chemistry*, 10(8), 982–1012. Retrieved 2012-11-12, from <http://onlinelibrary.wiley.com/doi/10.1002/jcc.540100804/abstract> doi: 10.1002/jcc.540100804
- Conte, D., Foggia, P., Sansone, C., & Vento, M. (2004). Thirty Years of Graph Matching in Pattern Recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(03), 265–298. Retrieved 2013-10-29, from <http://www.worldscientific.com/doi/abs/10.1142/S0218001404003228> doi: 10.1142/S0218001404003228
- Conte, D., Foggia, P., & Vento, M. (2007). Challenging Complexity of Maximum Common Subgraph Detection Algorithms: A Performance Analysis of Three Algorithms on a Wide Database of Graphs. *Journal of Graph Algorithms and Applications*, 11(1), 99–143. Retrieved 2014-09-12, from <http://www.emis.ams.org/journals/JGAA/accepted/2007/ConteFoggiaVento2007.11.1.pdf>
- Cook, S. A. (1971). The Complexity of Theorem-Proving Procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing* (pp. 151–158). New York, NY, USA: ACM. Retrieved 2012-11-08, from <http://doi.acm.org/10.1145/800157.805047> doi: 10.1145/800157.805047
- Cordella, L., Foggia, P., Sansone, C., & Vento, M. (1999). Performance evaluation of the VF graph matching algorithm. In *International Conference on Image Analysis and Processing, 1999. Proceedings* (pp. 1172–1177). Retrieved 2013-07-09, from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=797762 doi: 10.1109/ICIAP.1999.797762
- Cordella, L., Foggia, P., Sansone, C., & Vento, M. (2001). An Improved Algorithm for Matching Large Graphs. In *In: 3rd IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition, Cuen* (pp. 149–159).
- Cramer, R. (2012). R-group Template CoMFA Combines Benefits of "ad hoc" and Topomer Alignments using 3d-QSAR for Lead Optimization. *Journal of Computer-Aided Molecular Design*, 26(7), 805–819. Retrieved 2013-06-17, from <http://link.springer.com/article/10.1007%2Fs10822-012-9583-9> doi: 10.1007/s10822-012-9583-9

- Cross, S., Baroni, M., Carosati, E., Benedetti, P., & Clementi, S. (2010). FLAP: GRID Molecular Interaction Fields in Virtual Screening. Validation using the DUD Data Set. *Journal of Chemical Information and Modeling*, 50(8), 1442–1450. Retrieved 2013-09-24, from <http://dx.doi.org/10.1021/ci100221g> doi: 10.1021/ci100221g
- Crowe, J. E., Lynch, M. F., & Town, W. G. (1970). Analysis of Structural Characteristics of Chemical Compounds in a Large Computer-Based File. Part I. Non-Cyclic Fragments. *Journal of the Chemical Society C: Organic*(7), 990–996. Retrieved 2012-12-11, from <http://pubs.rsc.org/en/content/articlelanding/1970/j3/j39700000990> doi: 10.1039/J39700000990
- Dalby, A., Nourse, J. G., Hounshell, W. D., Gushurst, A. K. I., Grier, D. L., Leland, B. A., & Laufer, J. (1992). Description of Several Chemical Structure File Formats Used by Computer Programs Developed at Molecular Design Limited. *Journal of Chemical Information and Computer Sciences*, 32(3), 244–255. Retrieved 2012-11-07, from <http://dx.doi.org/10.1021/ci00007a012> doi: 10.1021/ci00007a012
- Dalke, A. (2012). *Topologically Non-Planar Compounds* [Content Site]. Retrieved 2015-03-20, from http://dalkescientific.com/writings/diary/archive/2012/05/18/nonplanar_compounds.html
- Dalke, A. (2013). *Varkony Reconsidered* [Corporate Website]. Retrieved 2014-09-23, from http://www.dalkescientific.com/writings/diary/archive/2013/07/27/varkony_reconsidered.html
- Dalke, A., & Hastings, J. (2013). FMCS: a Novel Algorithm for the Multiple MCS Problem. *Journal of Cheminformatics*, 5(1), 1–1. Retrieved 2014-09-19, from <http://link.springer.com/article/10.1186/1758-2946-5-S1-06> doi: 10.1186/1758-2946-5-S1-06
- Daylight Chemical Information Systems, f. (2011). *Daylight Theory: Fingerprints*. Retrieved 2013-01-06, from <http://www.daylight.com/dayhtml/doc/theory/theory.finger.html>
- Depolli, M., Konc, J., Rozman, K., Trobec, R., & Janežič, D. (2013). Exact Parallel Maximum Clique Algorithm for General and Protein Graphs. *Journal of Chemical Information and Modeling*, 53(9), 2217–2228. Retrieved 2013-11-07, from <http://dx.doi.org/10.1021/ci4002525> doi: 10.1021/ci4002525
- Diestel, R. (2006). *Graph Theory* (3rd ed.). New York: Springer-Verlag.
- Dossetter, A. G., Griffen, E. J., & Leach, A. G. (2013). Matched Molecular Pair Analysis in Drug Discovery. *Drug Discovery Today*, 18(15–16), 724–731. Retrieved 2013-11-27, from <http://www.sciencedirect.com/science/article/pii/S1359644613000937> doi: 10.1016/j.drudis.2013.03.003
- Dubois, J. E. (1973). French National Policy for Chemical Information and the DARC System as a Potential Tool of This Policy. *Journal of Chemical Documentation*, 13(1), 8–13. Retrieved 2013-01-10, from <http://dx.doi.org/10.1021/c160048a004> doi: 10.1021/c160048a004
- Dubois, J. E., Laurent, D., & Aranda, A. (1973). Méthode de Perturbation d'Environnements Limites Con-

- centriques Ordonnés (PELCO). *Journal de Chimie Physique et de Physico-Chimie Biologique*, 70, 1608–1615.
- Dubois, J. E., Panaye, A., & Attias, R. (1987). DARC System: Notions of Defined and Generic Substructures. Filiation and Coding of FREL Substructure (SS) Classes. *Journal of Chemical Information and Computer Sciences*, 27(2), 74–82. Retrieved 2013-01-10, from <http://dx.doi.org/10.1021/ci00054a007> doi: 10.1021/ci00054a007
- Duesbury, E., Holliday, J., & Willett, P. (2015). Maximum Common Substructure-Based Data Fusion in Similarity Searching. *Journal of Chemical Information and Modeling*, 55(2), 222–230. doi: 10.1021/ci5005702
- Durand, P. J., Pasari, R., Baker, J. W., & Tsai, C. (1999). An Efficient Algorithm for Similarity Analysis of Molecules. *Internet Journal of Chemistry*, 2(17), 1–16. Retrieved 2012-11-22, from <http://www.cs.kent.edu/~jbaker/paper/>
- Durant, J. L., Leland, B. A., Henry, D. R., & Nourse, J. G. (2002). Reoptimization of MDL Keys for Use in Drug Discovery. *Journal of Chemical Information and Computer Sciences*, 42(6), 1273–1280. Retrieved 2012-11-06, from <http://dx.doi.org/10.1021/ci010132r> doi: 10.1021/ci010132r
- Dyson, G., Cossum, W., Lynch, M., & Morgan, H. (1963). Mechanical Manipulation of Chemical Structure: Molform Computation and Substructure Searching of Organic Structures by the Use of Cipherdirected, Extended and Random Matrices. *Inform. Storage Ret.*, 1(2–3), 69–99. Retrieved 2013-08-27, from <http://www.sciencedirect.com/science/article/pii/0020027163900111> doi: 10.1016/0020-0271(63)90011-1
- Ehrlich, H.-C., & Rarey, M. (2011). Maximum Common Subgraph Isomorphism Algorithms and Their Applications in Molecular Science: A Review. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 1(1), 68–79. Retrieved 2014-10-24, from <http://onlinelibrary.wiley.com/doi/10.1002/wcms.5/abstract> doi: 10.1002/wcms.5
- Ehrlich, H.-C., & Rarey, M. (2012). Systematic Benchmark of Substructure Search in Molecular Graphs - from Ullmann to VF2. *Journal of Cheminformatics*, 4(1), 13. Retrieved 2012-11-30, from <http://www.jcheminf.com/content/4/1/13/abstract> doi: 10.1186/1758-2946-4-13
- Ekins, S., Honeycutt, J. D., & Metz, J. T. (2010). Evolving Molecules Using Multi-Objective Optimization: Applying to Adme/Tox. *Drug Discovery Today*, 15(11–12), 451–460. Retrieved 2013-09-24, from <http://www.sciencedirect.com/science/article/pii/S1359644610001467> doi: 10.1016/j.drudis.2010.04.003
- Englert, P., & Kovács, P. (2015). Efficient Heuristics for Maximum Common Substructure Search. *Journal of Chemical Information and Modeling*, 55(5), 941–955. Retrieved 2015-09-23, from <http://dx.doi.org/10.1021/acs.jcim.5b00036> doi: 10.1021/acs.jcim.5b00036
- Ersmark, K., Feierberg, I., Bjelic, S., Hultén, J., Samuelsson, B., Åqvist, J., & Hallberg, A. (2003). C2-Symmetric Inhibitors of Plasmodium falciparum Plasmeprin II: Synthesis and Theoretical Pre-

- dictions. *Bioorganic & Medicinal Chemistry*, 11(17), 3723–3733. Retrieved 2015-03-07, from <http://www.sciencedirect.com/science/article/pii/S0968089603003390> doi: 10.1016/S0968-0896(03)00339-0
- Euler, L. (1741). *Solutio Problematis ad Geometriam Situs Pertinentis*. *Commentarii academiae scientiarum Petropolitanae*, 8, 128–140. Retrieved 2012-12-07, from <http://www.math.dartmouth.edu/~euler/docs/originals/E053.pdf>
- Evans, A., Wilson, R., & Hancock, E. (1995). Matching SAR Images Using a Hierarchical Constraint Process. In *Fifth International Conference on Image Processing and its Applications, 1995* (pp. 60–64). doi: 10.1049/cp:19950620
- Faulon, J.-L. (1998). Isomorphism, Automorphism Partitioning, and Canonical Labeling Can Be Solved in Polynomial-Time for Molecular Graphs. *Journal of Chemical Information and Computer Sciences*, 38(3), 432–444. Retrieved 2015-03-23, from <http://pubs.acs.org/doi/abs/10.1021/ci9702914> doi: 10.1021/ci9702914
- Feher, M. (2006). Consensus Scoring for Protein–Ligand Interactions. *Drug Discovery Today*, 11(9–10), 421–428. Retrieved 2013-09-18, from <http://www.sciencedirect.com/science/article/pii/S1359644606000523> doi: 10.1016/j.drudis.2006.03.009
- Ferrer, M., Valveny, E., Serratos, F., Riesen, K., & Bunke, H. (2010). Generalized Median Graph Computation by Means of Graph Embedding in Vector Spaces. *Pattern Recognition*, 43(4), 1642–1655. Retrieved 2012-11-05, from <http://www.sciencedirect.com/science/article/pii/S0031320309003860> doi: 10.1016/j.patcog.2009.10.013
- Floyd, R. W. (1962). Algorithm 97: Shortest Path. *Commun. ACM*, 5(6), 345–345. Retrieved 2015-03-24, from <http://doi.acm.org/10.1145/367766.368168> doi: 10.1145/367766.368168
- Franco, P., Porta, N., Holliday, J. D., & Willett, P. (2014). The Use of 2d Fingerprint Methods to Support the Assessment of Structural Similarity in Orphan Drug Legislation. *Journal of Cheminformatics*, 6(1), 5. Retrieved 2015-03-20, from <http://www.jcheminf.com/content/6/1/5/abstract> doi: 10.1186/1758-2946-6-5
- Free, S. M., & Wilson, J. W. (1964). A Mathematical Contribution to Structure-Activity Studies. *Journal of Medicinal Chemistry*, 7(4), 395–399. Retrieved 2012-12-12, from <http://dx.doi.org/10.1021/jm00334a001> doi: 10.1021/jm00334a001
- Fruchterman, T. M. J., & Reingold, E. M. (1991). Graph Drawing by Force-Directed Placement. *Software: Practice and Experience*, 21(11), 1129–1164. Retrieved 2013-11-26, from <http://onlinelibrary.wiley.com/doi/10.1002/spe.4380211102/abstract> doi: 10.1002/spe.4380211102
- Gardiner, E. J., Artymiuk, P. J., & Willett, P. (1997). Clique-Detection Algorithms for Matching Three-Dimensional Molecular Structures. *Journal of Molecular Graphics and Modelling*, 15(4), 245–253. Retrieved 2012-11-26, from <http://ukpmc.ac.uk/abstract/MED/9524934>
- Gardiner, E. J., Gillet, V. J., Haranczyk, M., Hert, J., Holliday, J. D., Malim, N., . . . Willett, P. (2009). Turbo

- Similarity Searching: Effect of Fingerprint and Dataset on Virtual-Screening Performance. *Statistical Analysis and Data Mining*, 2(2), 103–114. Retrieved 2013-08-16, from <http://onlinelibrary.wiley.com/doi/10.1002/sam.10037/abstract> doi: 10.1002/sam.10037
- Garey, M. R., & Johnson, D. S. (1979). *Computers and Intractability*. San Francisco, Calif.: W. H. Freeman and Co. Retrieved 2012-11-08, from <http://www.ams.org/mathscinet-getitem?mr=519066> (A guide to the theory of NP-completeness)
- Gaulton, A., Bellis, L. J., Bento, A. P., Chambers, J., Davies, M., Hersey, A., ... Overington, J. P. (2011). ChEMBL: A Large-Scale Bioactivity Database for Drug Discovery. *Nucleic Acids Research*. Retrieved 2012-11-07, from <http://nar.oxfordjournals.org/content/early/2011/09/22/nar.gkr777> doi: 10.1093/nar/gkr777
- Gillet, V. J., Downs, G. M., Ling, A., Lynch, M. F., Venkataram, P., Wood, J. V., & Dethlefsen, W. (1987). 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*, 27(3), 126–137. Retrieved 2016-01-30, from <http://dx.doi.org/10.1021/ci00055a007> doi: 10.1021/ci00055a007
- Gillet, V. J., Khatib, W., Willett, P., Fleming, P. J., & Green, D. V. S. (2002). Combinatorial Library Design Using a Multiobjective Genetic Algorithm. *Journal of Chemical Information and Computer Sciences*, 42(2), 375–385. Retrieved 2013-09-24, from <http://dx.doi.org/10.1021/ci010375j> doi: 10.1021/ci010375j
- Goetz, B. (2004). Java Theory and Practice: Dynamic Compilation and Performance Measurement. *Java Theory and Practice*, 1–12. Retrieved 2015-07-17, from <http://www.ibm.com/developerworks/java/library/j-jtp12214/j-jtp12214-pdf.pdf>
- Goetz, B. (2005). Java Theory and Practice: Anatomy of a Flawed Microbenchmark. *Java Theory and Practice*, 1–9. Retrieved 2015-07-17, from <https://www.ibm.com/developerworks/java/library/j-jtp02225/j-jtp02225-pdf.pdf>
- Good, A. C., & Oprea, T. I. (2008). Optimization of CAMD Techniques 3. Virtual Screening Enrichment Studies: A Help or Hindrance in Tool Selection? *Journal of Computer-Aided Molecular Design*, 22(3-4), 169–178. Retrieved 2015-09-02, from <http://link.springer.com/article/10.1007/s10822-007-9167-2> doi: 10.1007/s10822-007-9167-2
- Gouda, K., & Hassaan, M. (2012). A Fast Algorithm for Subgraph Search Problem. In *2012 8th International Conference on Informatics and Systems, INFOS 2012* (pp. DE53–DE58).
- Griffen, E., Leach, A., Robb, G., & Warner, D. (2011). Matched Molecular Pairs as a Medicinal Chemistry Tool. *Journal of Medicinal Chemistry*, 54(22), 7739–7750. Retrieved 2013-06-17, from <http://pubs.acs.org/doi/abs/10.1021/jm200452d> doi: 10.1021/jm200452d
- Grosso, A., Locatelli, M., & Pullan, W. (2008). Simple Ingredients Leading to Very Efficient Heuristics for the Maximum Clique Problem. *Journal of Heuristics*, 14(6), 587–612. Retrieved 2013-08-16, from

- <http://dx.doi.org/10.1007/s10732-007-9055-x> doi: 10.1007/s10732-007-9055-x
- Guha, R. (2012). Exploring Structure–Activity Data Using the Landscape Paradigm. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, *2*(6), 829–841. Retrieved 2013-09-25, from <http://onlinelibrary.wiley.com/doi/10.1002/wcms.1087/abstract> doi: 10.1002/wcms.1087
- Guha, R., & Van Drie, J. H. (2008). Structure-Activity Landscape Index: Identifying and Quantifying Activity Cliffs. *Journal of Chemical Information and Modeling*, *48*(3), 646–658. Retrieved 2013-11-23, from <http://dx.doi.org/10.1021/ci7004093> doi: 10.1021/ci7004093
- Hagadone, T. R. (1992). Molecular Substructure Similarity Searching: Efficient Retrieval in Two-Dimensional Structure Databases. *Journal of Chemical Information and Computer Sciences*, *32*(5), 515–521. Retrieved 2012-11-06, from <http://dx.doi.org/10.1021/ci00009a019> doi: 10.1021/ci00009a019
- Hall, D., & Llinas, J. (1997). An Introduction to Multisensor Data Fusion. *Proceedings of the IEEE*, *85*(1), 6–23. Retrieved from <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=554205> doi: 10.1109/5.554205
- Han, L., Wilson, R., & Hancock, E. (2015). Generative Graph Prototypes from Information Theory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *37*(10), 2013–2027. doi: 10.1109/TPAMI.2015.2400451
- Haranczyk, M., & Holliday, J. (2008). Comparison of Similarity Coefficients for Clustering and Compound Selection. *Journal of Chemical Information and Modeling*, *48*(3), 498–508. Retrieved 2013-09-11, from <http://dx.doi.org/10.1021/ci700413a> doi: 10.1021/ci700413a
- Hariharan, R., Janakiraman, A., Nilakantan, R., Singh, B., Varghese, S., Landrum, G., & Schuffenhauer, A. (2011). MultiMCS: A Fast Algorithm for the Maximum Common Substructure Problem on Multiple Molecules. *Journal of Chemical Information and Modeling*, *51*(4), 788–806. Retrieved 2015-07-28, from <http://dx.doi.org/10.1021/ci100297y> doi: 10.1021/ci100297y
- Håstad, J. (1999). Clique Is Hard to Approximate Within $N^{1-\epsilon}$. *Acta Mathematica*, *182*(1), 105–142. Retrieved 2012-12-12, from <http://link.springer.com/article/10.1007/BF02392825> doi: 10.1007/BF02392825
- He, L., & Jurs, P. C. (2005). Assessing the Reliability of a QSAR Model's Predictions. *Journal of Molecular Graphics and Modelling*, *23*(6), 503–523. Retrieved 2013-06-14, from <http://www.sciencedirect.com/science/article/pii/S1093326305000173> doi: 10.1016/j.jmkgm.2005.03.003
- Heller, S. R., & McNaught, A. D. (2009). The IUPAC International Chemical Identifier (InChI). *Chemistry International*, *31*(1), 7–12. Retrieved 2012-11-07, from <http://www.freepatentsonline.com/article/Chemistry-International/192975220.html>
- Henikoff, S., & Henikoff, J. G. (1992). Amino Acid Substitution Matrices from Protein Blocks. *Proceedings of the National Academy of Sciences of the United States of America*, *89*(22), 10915–10919. Retrieved 2015-06-16, from <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC50453/>
- Hert, J., Willett, P., Wilton, D. J., Acklin, P., Azzaoui, K., Jacoby, E., & Schuffenhauer, A. (2004). Com-

- parison of Topological Descriptors for Similarity-Based Virtual Screening Using Multiple Bioactive Reference Structures. *Organic & Biomolecular Chemistry*, 2(22), 3256. Retrieved 2013-07-08, from <http://pubs.rsc.org/en/content/articlehtml/2004/ob/b409865j> doi: 10.1039/b409865j
- Hert, J., Willett, P., Wilton, D. J., Acklin, P., Azzaoui, K., Jacoby, E., & Schuffenhauer, A. (2005). Enhancing the Effectiveness of Similarity-Based Virtual Screening Using Nearest-Neighbor Information. *Journal of Medicinal Chemistry*, 48(22), 7049–7054. Retrieved 2013-11-13, from <http://dx.doi.org/10.1021/jm050316n> doi: 10.1021/jm050316n
- Hert, J., Willett, P., Wilton, D. J., Acklin, P., Azzaoui, K., Jacoby, E., & Schuffenhauer, A. (2006). New Methods for Ligand-Based Virtual Screening: Use of Data Fusion and Machine Learning to Enhance the Effectiveness of Similarity Searching. *Journal of Chemical Information and Modeling*, 46(2), 462–470. Retrieved 2013-11-13, from <http://dx.doi.org/10.1021/ci050348j> doi: 10.1021/ci050348j
- Hintze, J. L., & Nelson, R. D. (1998). Violin Plots: A Box Plot-Density Trace Synergism. *The American Statistician*, 52(2), 181–184. Retrieved 2016-02-27, from <http://www.tandfonline.com/doi/abs/10.1080/00031305.1998.10480559> doi: 10.1080/00031305.1998.10480559
- Hiz, H. (1964). A Linearization of Chemical Graphs. *Journal of Chemical Documentation*, 4(3), 173–180. Retrieved 2012-11-07, from <http://dx.doi.org/10.1021/c160014a015> doi: 10.1021/c160014a015
- Hlaoui, A., & Wang, S. (2006). Median Graph Computation for Graph Clustering. *Soft Comput.*, 10(1), 47–53. Retrieved 2012-11-05, from <http://dx.doi.org/10.1007/s00500-005-0464-1> doi: 10.1007/s00500-005-0464-1
- Holliday, J. D., Hu., C., & Willett, P. (2002). Grouping of Coefficients for the Calculation of Inter-Molecular Similarity and Dissimilarity using 2d Fragment Bit-Strings. *Combinatorial Chemistry & High Throughput Screening*, 5(2). Retrieved 2013-07-04, from <http://eprints.whiterose.ac.uk/7690/> doi: 10.2174/1386207024607338
- Holliday, J. D., Kanoulas, E., Malim, N., & Willett, P. (2011). Multiple Search Methods for Similarity-Based Virtual Screening: Analysis of Search Overlap and Precision. *Journal of Cheminformatics*, 3(1), 29. Retrieved 2013-11-12, from <http://www.jcheminf.com/content/3/1/29/abstract> doi: 10.1186/1758-2946-3-29
- Holliday, J. D., Salim, N., Whittle, M., & Willett, P. (2003). Analysis and Display of the Size Dependence of Chemical Similarity Coefficients. *Journal of Chemical Information and Computer Sciences*, 43(3), 819–828. Retrieved 2013-03-12, from <http://dx.doi.org/10.1021/ci034001x> doi: 10.1021/ci034001x
- Holliday, J. D., Willett, P., & Xiang, H. (2012). Interactions Between Weighting Scheme and Similarity Coefficient in Similarity-Based Virtual Screening. *International Journal of Chemoinformatics and Chemical Engineering*, 2(2), 28–41. Retrieved 2013-08-05, from <http://www.igi-global.com/article/interactions-between-weighting-scheme-similarity/68019>
- Horvath, D., Marcou, G., & Varnek, A. (2013). Do Not Hesitate to Use Tversky—and Other Hints for

- Successful Active Analogue Searches with Feature Count Descriptors. *Journal of Chemical Information and Modeling*. Retrieved 2013-06-18, from <http://dx.doi.org/10.1021/ci400106g> doi: 10.1021/ci400106g
- Hu, X., Hu, Y., Vogt, M., Stumpfe, D., & Bajorath, J. (2012). MMP-Cliffs: Systematic Identification of Activity Cliffs on the Basis of Matched Molecular Pairs. *Journal of Chemical Information and Modeling*, 52(5), 1138–1145. Retrieved 2013-11-27, from <http://dx.doi.org/10.1021/ci3001138> doi: 10.1021/ci3001138
- Hu, Y., Stumpfe, D., & Bajorath, J. (2011). Lessons Learned from Molecular Scaffold Analysis. *Journal of Chemical Information and Modeling*, 51(8), 1742–1753. Retrieved 2013-11-26, from <http://europepmc.org/abstract/MED/21755989> doi: 10.1021/ci200179y
- Huang, C., Xu, Q., Chen, C., Song, C., Xu, Y., Xiang, Y., ... Jiang, H. (2014). The Rapid Discovery and Identification of Physalins in the Calyx of *Physalis Alkekengi* L.var.franchetii (mast.) Makino Using Ultra-High Performance Liquid Chromatography–Quadrupole Time of Flight Tandem Mass Spectrometry Together with a Novel Three-Step Data Mining Strategy. *Journal of Chromatography A*, 1361, 139–152. Retrieved 2015-03-19, from <http://www.sciencedirect.com/science/article/pii/S0021967314012321> doi: 10.1016/j.chroma.2014.08.004
- Huang, X., Lai, J., & Jennings, S. F. (2006). Maximum Common Subgraph: Some Upper Bound and Lower Bound Results. *BMC Bioinformatics*, 7(Suppl 4), S6. Retrieved 2015-08-22, from <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1780128/> doi: 10.1186/1471-2105-7-S4-S6
- Hull, S. E., Barnard, J. M., & Thomas, D. G. (2011). *InChI Source Code Documentation*. Digital Chemistry Ltd. Retrieved 2013-01-09, from http://www.inchi-trust.org/fileadmin/user_upload/software/inchi-v1.03/InChI_Source_Code_Documentation_v1.0.pdf
- Iyer, P., Stumpfe, D., Vogt, M., Bajorath, J., & Maggiora, G. M. (2013). Activity Landscapes, Information Theory, and Structure – Activity Relationships. *Molecular Informatics*, 32(5-6), 421–430. Retrieved 2013-11-20, from <http://onlinelibrary.wiley.com/doi/10.1002/minf.201200120/abstract> doi: 10.1002/minf.201200120
- Jaccard, P. (1908). Nouvelles Recherches sur La Distribution Florale. *Bulletin de la Société Vaudense des Sciences Naturelles*, 44, 223–270. Retrieved 2013-06-17, from <http://www.citeulike.org/user/echinotrix/article/7707101>
- James, C. A., Weininger, D., & Delany, J. (2004). *Daylight Theory: Fingerprints*. Retrieved 2012-12-20, from <http://www.daylight.com/dayhtml/doc/theory/theory.finger.html>
- Jeliazkova, N., & Kochev, N. (2011). AMBIT-SMARTS: Efficient Searching of Chemical Structures and Fragments. *Molecular Informatics*, 30(8), 707–720. Retrieved 2015-03-10, from <http://onlinelibrary.wiley.com/doi/10.1002/minf.201100028/abstract> doi: 10.1002/minf.201100028
- Jiang, X., Müunger, A., & Bunke, H. (2001). On Median Graphs: Properties, Algorithms, and Applications. *IEEE Transactions. Pattern Analysis and Machine Intelligence*, 23(10), 1144–1151. Retrieved 2012-11-

- 05, from <http://dx.doi.org/10.1109/34.954604> doi: 10.1109/34.954604
- Johnson, M., Basak, S., & Maggiora, G. (1988). A Characterization of Molecular Similarity Methods for Property Prediction. *Mathematical and Computer Modelling*, 11, 630–634. Retrieved 2013-01-07, from <http://www.sciencedirect.com/science/article/pii/0895717788905699> doi: 10.1016/0895-7177(88)90569-9
- Johnson, M. A., & Maggiora, G. (1990). *Concepts and Applications of Molecular Similarity*. New York: Wiley. Retrieved 2012-12-10, from <http://www.amazon.com/Concepts-Applications-Molecular-Similarity-Johnson/dp/0471621757>
- Kariba, R. M., Houghton, P. J., & Yenesew, A. (2002). Antimicrobial Activities of a New Schizozygane Indoline Alkaloid from *Schizozygia Coffaeoides* and the Revised Structure of Isoschizogaline. *Journal of Natural Products*, 65(4), 566–569. Retrieved from <http://pubs.acs.org/doi/full/10.1021/np010298m> doi: 10.1021/np010298m
- Kawabata, T. (2011). Build-Up Algorithm for Atomic Correspondence between Chemical Structures. *Journal of Chemical Information and Modeling*, 51(8), 1775–1787. Retrieved 2014-09-17, from <http://dx.doi.org/10.1021/ci2001023> doi: 10.1021/ci2001023
- Kawabata, T., & Nakamura, H. (2014). 3d Flexible Alignment Using 2d Maximum Common Substructure: Dependence of Prediction Accuracy on Target-Reference Chemical Similarity. *Journal of Chemical Information and Modeling*, 54(7), 1850–63. Retrieved 2014-06-12, from <http://dx.doi.org/10.1021/ci500006d> doi: 10.1021/ci500006d
- Kearsley, S. K., Sallamack, S., Fluder, E. M., Andose, J. D., Mosley, R. T., & Sheridan, R. P. (1996). Chemical Similarity Using Physicochemical Property Descriptors. *Journal of Chemical Information and Computer Sciences*, 36(1), 118–127. Retrieved 2013-09-19, from <http://dx.doi.org/10.1021/ci950274j> doi: 10.1021/ci950274j
- Keiser, M. J., Setola, V., Irwin, J. J., Laggner, C., Abbas, A. I., Hufeisen, S. J., ... Roth, B. L. (2009). Predicting New Molecular Targets for Known Drugs. *Nature*, 462(7270), 175–181. Retrieved 2013-06-14, from <http://www.nature.com/nature/journal/v462/n7270/abs/nature08506.html> doi: 10.1038/nature08506
- Klinger, S., & Austin, J. (2005). Chemical Similarity Searching Using a Neural Graph Matcher. In *ESANN 2005 Proceedings* (pp. 479–484). Bruges, Belgium. Retrieved 2015-02-28, from <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.92.5082&rep=rep1&type=pdf>
- Klinger, S., & Austin, J. (2006). Weighted Superstructures for Chemical Similarity Searching. In *Proceedings of the 9th Joint Conference on Information Sciences*. Kaohsiung, Taiwan: Atlantis Press. Retrieved 2015-01-07, from http://www.google.co.uk/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CCcQFjAA&url=http%3A%2F%2Fwww.cs.york.ac.uk%2Farch%2Fpublications%2Fbyyear%2F2006%2F2006_WeightedSuperstructuresForChemicalSimilarity_234

- .pdf&ei=p02tVJSOBsH1aMC7gPgF&usg=AFQjCNHPDpU1-3V_ctj1epYKp0EeEt_-oA&sig2=yI2JmENskck7Cqtp5piINg&bvm=bv.83134100,d.d2s
- Koch, I. (2001). Enumerating All Connected Maximal Common Subgraphs in Two Graphs. *Theoretical Computer Science*, 250(1–2), 1–30. Retrieved 2012-10-11, from <http://www.sciencedirect.com/science/article/pii/S0304397500002863> doi: 10.1016/S0304-3975(00)00286-3
- Kovács, P., & Englert, P. (2013). *MaxCommonSubstructure (JChem API documentation (c) 1998-2013 ChemAxon Ltd.)*. Retrieved 2013-08-16, from <http://www.chemaxon.com/jchem/doc/dev/java/api/com/chemaxon/search/mcs/MaxCommonSubstructure.html>
- Kozen, D. (1978). A Clique Problem Equivalent to Graph Isomorphism. *SIGACT News*, 10(2), 50–52. Retrieved 2012-11-21, from <http://doi.acm.org/10.1145/990524.990529> doi: 10.1145/990524.990529
- Krasowski, M. D., Siam, M. G., Iyer, M., & Ekins, S. (2009). Molecular Similarity Methods for Predicting Cross-Reactivity With Therapeutic Drug Monitoring Immunoassays. *Therapeutic Drug Monitoring*, 31(3), 337–344. Retrieved 2012-11-06, from <http://europepmc.org/articles/PMC2846282> doi: 10.1097/FTD.0b013e31819c1b83
- Kruskal, J. B. (1956). On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *Proceedings of the American Mathematical Society*, 7(1), 48–50. Retrieved 2012-11-28, from <http://www.jstor.org/stable/10.2307/2033241>
- Landrum, G. (2012). *Fingerprints in the RDKit* [Slideshow]. London. Retrieved 2013-11-28, from http://rdkit.org/UGM/2012/Landrum_RDKit_UGM.Fingerprints.Final.pptx.pdf
- Leach, A. G., Jones, H. D., Cosgrove, D. A., Kenny, P. W., Ruston, L., MacFaul, P., ... Law, B. (2006). 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*, 49(23), 6672–6682. Retrieved 2013-11-27, from <http://dx.doi.org/10.1021/jm0605233> doi: 10.1021/jm0605233
- Leach, A. R., & Gillet, V. J. (2007a). Appendix 1: Matrices, Eigenvectors and Eigenvalues. In *An Introduction to Chemoinformatics* (2nd ed., p. 203). Springer.
- Leach, A. R., & Gillet, V. J. (2007b). Chapter 3: Molecular Descriptors. In *An Introduction to Chemoinformatics* (2nd ed., pp. 53–77). Springer.
- Leach, A. R., Gillet, V. J., Lewis, R. A., & Taylor, R. (2010). Three-Dimensional Pharmacophore Methods in Drug Discovery. *Journal of Medicinal Chemistry*, 53(2), 539–558. Retrieved 2012-11-14, from <http://dx.doi.org/10.1021/jm900817u> doi: 10.1021/jm900817u
- Li, L.-M., Li, G.-Y., Ding, L.-S., Lei, C., Yang, L.-B., Zhao, Y., ... Sun, H.-D. (2007). Sculponins A–C, three new 6,7-seco-ent-kauranoids from *Isodon sculponeatus*. *Tetrahedron Letters*, 48(52), 9100–9103. Retrieved 2015-03-19, from <http://www.sciencedirect.com/science/article/pii/S004040390702148X> doi: 10.1016/j.tetlet.2007.10.133

- Libby, P. R., Munson, B. R., Fiel, R. J., & Porter, C. W. (1995). Cationic Porphyrin Derivatives as Inhibitors of Polyamine Catabolism. *Biochemical Pharmacology*, 50(9), 1527–1530. Retrieved 2015-03-17, from <http://www.sciencedirect.com/science/article/pii/S0006295295020667> doi: 10.1016/0006-2952(95)02066-7
- Lin, D. (1998). An Information-Theoretic Definition of Similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning* (pp. 296–304). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. Retrieved 2013-06-12, from <http://webdocs.cs.ualberta.ca/~lindek/papers/sim.pdf>
- Lounkine, E., Keiser, M. J., Whitebread, S., Mikhailov, D., Hamon, J., Jenkins, J. L., ... Urban, L. (2012). Large-Scale Prediction and Testing of Drug Activity on Side-Effect Targets. *Nature*, 486(7403), 361–367. Retrieved 2013-06-14, from <http://www.nature.com/nature/journal/v486/n7403/abs/nature11159.html> doi: 10.1038/nature11159
- Lucas, S., Heim, R., Negri, M., Antes, I., Ries, C., Schewe, K. E., ... Hartmann, R. W. (2008). Novel Aldosterone Synthase Inhibitors with Extended Carbocyclic Skeleton by a Combined Ligand-Based and Structure-Based Drug Design Approach. *Journal of Medicinal Chemistry*, 51(19), 6138–6149. Retrieved 2012-11-14, from <http://dx.doi.org/10.1021/jm800683c>
- Mackey, M. D., & Melville, J. L. (2009). Better than Random? The Chemotype Enrichment Problem. *Journal of Chemical Information and Modeling*, 49(5), 1154–1162. Retrieved 2014-06-21, from <http://dx.doi.org/10.1021/ci8003978> doi: 10.1021/ci8003978
- Maggiora, G. M. (2006). On Outliers and Activity Cliffs - Why QSAR Often Disappoints. *Journal of Chemical Information and Modeling*, 46(4), 1535–1535. Retrieved 2013-11-19, from <http://dx.doi.org/10.1021/ci060117s> doi: 10.1021/ci060117s
- Maurer, P. J., & Rapoport, H. (1987). Nitrogen-Bridged Conformationally Constrained Etorphine Analogues. Synthesis and Biological Evaluation. *Journal of Medicinal Chemistry*, 30(11), 2016–2026.
- May, J. (2013). *Efficient Bits: New SMILES behaviour - parsing (CDK 1.5.4)* [Blog]. Retrieved 2016-02-06, from <http://efficientbits.blogspot.co.uk/2013/12/new-smiles-behaviour-parsing-cdk-154.html>
- May, J., & Steinbeck, C. (2014). Efficient Ring Perception for the Chemistry Development Kit. *Journal of Cheminformatics*, 6(1), 3. Retrieved 2015-03-21, from <http://www.jcheminf.com/content/6/1/3/abstract> doi: 10.1186/1758-2946-6-3
- McGregor, J. J. (1982). Backtrack Search Algorithms and the Maximal Common Subgraph Problem. *Software: Practice and Experience*, 12(1), 23–34. Retrieved 2012-10-19, from <http://onlinelibrary.wiley.com/doi/10.1002/spe.4380120103/abstract> doi: 10.1002/spe.4380120103
- McGregor, J. J., & Willett, P. (1981). Use of a Maximum Common Subgraph Algorithm in the Automatic Identification of Ostensible Bond Changes Occurring in Chemical Reactions. *Journal of Chemical Information and Computer Sciences*, 21(3), 137–140. Retrieved 2013-08-29, from <http://dx.doi.org/>

- 10.1021/ci00031a005 doi: 10.1021/ci00031a005
- Menon, G. K., & Cammarata, A. (1977). Pattern Recognition II: Investigation of Structure-Activity Relationships. *Journal of Pharmaceutical Sciences*, 66(3), 304–314. Retrieved 2013-10-03, from <http://onlinelibrary.wiley.com/doi/10.1002/jps.2600660303/abstract> doi: 10.1002/jps.2600660303
- Mercier, C., Fabart, V., Sobel, Y., & Dubois, J. E. (1991). Modeling Alcohol Metabolism with the DARC/CALPHI System. *Journal of Medicinal Chemistry*, 34(3), 934–942. Retrieved 2013-10-03, from <http://pubs.acs.org/doi/pdf/10.1021/jm00107a010>
- Mestres, J., & Maggiora, G. M. (2005). Putting Molecular Similarity into Context: Asymmetric Indices for Field-Based Similarity Measures. *Journal of Mathematical Chemistry*, 39(1), 107–118. Retrieved 2013-06-21, from <http://link.springer.com/content/pdf/10.1007%2Fs10910-005-9007-3.pdf> doi: 10.1007/s10910-005-9007-3
- Morgan, H. L. (1965). The Generation of a Unique Machine Description for Chemical Structures-A Technique Developed at Chemical Abstracts Service. *Journal of Chemical Documentation*, 5(2), 107–113. Retrieved 2012-11-07, from <http://dx.doi.org/10.1021/c160017a018> doi: 10.1021/c160017a018
- Myint, K.-Z., Wang, L., Tong, Q., & Xie, X.-Q. (2012). Molecular Fingerprint-Based Artificial Neural Networks QSAR for Ligand Biological Activity Predictions. *Molecular Pharmaceutics*, 9(10), 2912–2923. Retrieved 2013-10-25, from <http://dx.doi.org/10.1021/mp300237z> doi: 10.1021/mp300237z
- Navigli, R., & Ponzetto, S. P. (2012). Babelnet: The Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network. *Artificial Intelligence*, 193, 217–250. Retrieved 2012-11-01, from <http://www.sciencedirect.com/science/article/pii/S0004370212000793> doi: 10.1016/j.artint.2012.07.001
- Nebeský, L. (1971). Median Graphs. *Commentationes Mathematicae Universitatis Carolinae*, 12(2), 317–325. Retrieved 2012-11-05, from <http://dml.cz/dmlcz/105347>
- Nilakantan, R., Nunn, D. S., Greenblatt, L., Walker, G., Haraki, K., & Mobilio, D. (2006). A Family of Ring System-Based Structural Fragments for Use in Structure-Activity Studies: Database Mining and Recursive Partitioning. *Journal of Chemical Information and Modeling*, 46(3), 1069–1077. Retrieved 2014-08-01, from <http://dx.doi.org/10.1021/ci050521b> doi: 10.1021/ci050521b
- Nisius, B., Vogt, M., & Bajorath, J. (2009). Development of a Fingerprint Reduction Approach for Bayesian Similarity Searching Based on Kullback-Leibler Divergence Analysis. *Journal of Chemical Information and Modeling*, 49(6), 1347–1358. Retrieved 2013-08-05, from <http://dx.doi.org/10.1021/ci900087y> doi: 10.1021/ci900087y
- O'Boyle, N. M. (2012). Towards a Universal SMILES Representation - A Standard Method to Generate Canonical SMILES based on the InChI. *Journal of Cheminformatics*, 4(1), 22. Retrieved 2012-12-20, from <http://www.jcheminf.com/content/4/1/22/abstract> doi: 10.1186/1758-2946-4-22
- O'Boyle, N. M., Banck, M., James, C. A., Morley, C., Vandermeersch, T., & Hutchison, G. R. (2011). Open

- Babel: An Open Chemical Toolbox. *Journal of Cheminformatics*, 3(1), 33. Retrieved 2013-01-06, from <http://www.jcheminf.com/content/3/1/33/abstract> doi: 10.1186/1758-2946-3-33
- Owen, J. R., Nabney, I. T., Medina-Franco, J. L., & López-Vallejo, F. (2011). Visualization of Molecular Fingerprints. *Journal of Chemical Information and Modeling*, 51(7), 1552–1563. Retrieved 2013-01-09, from <http://dx.doi.org/10.1021/ci1004042> doi: 10.1021/ci1004042
- Palyulin, Radchenko, & Zefirov. (2000). Molecular Field Topology Analysis Method in QSAR Studies of Organic Compounds. *Journal of Chemical Information and Computer Sciences*, 40(3), 659–667.
- Papadatos, G., Cooper, A. W. J., Kadiramanathan, V., Macdonald, S. J. F., McLay, I. M., Pickett, S. D., ... Gillet, V. J. (2009). Analysis of Neighborhood Behavior in Lead Optimization and Array Design. *Journal of Chemical Information and Modeling*, 49(2), 195–208. Retrieved 2012-11-06, from <http://dx.doi.org/10.1021/ci800302g> doi: 10.1021/ci800302g
- Pardalos, P. M., & Rebennack, S. (2010). Computational Challenges with Cliques, Quasi-cliques and Clique Partitions in Graphs. In P. Festa (Ed.), *Experimental Algorithms* (Vol. 6049, pp. 13–22). Springer Berlin Heidelberg. Retrieved 2012-11-26, from http://link.springer.com/chapter/10.1007/978-3-642-13193-6_2
- Pardalos, P. M., & Xue, J. (1994). The Maximum Clique Problem. *Journal of Global Optimization*, 4(3), 301–328. Retrieved 2014-12-10, from <http://link.springer.com/article/10.1007/BF01098364> doi: 10.1007/BF01098364
- Peltason, L., Iyer, P., & Bajorath, J. (2010). Rationalizing Three-Dimensional Activity Landscapes and the Influence of Molecular Representations on Landscape Topology and the Formation of Activity Cliffs. *Journal of Chemical Information and Modeling*, 50(6), 1021–1033. Retrieved 2013-11-19, from <http://dx.doi.org/10.1021/ci100091e> doi: 10.1021/ci100091e
- Pizzirani, D., Roberti, M., Cavalli, A., Grimaudo, S., Di Cristina, A., Pipitone, R. M., ... Recanatini, M. (2008). Antiproliferative Agents That Interfere with the Cell Cycle at the G1 S Transition: Further Development and Characterization of a Small Library of Stilbene-Derived Compounds. *ChemMedChem*, 3(2), 345–355. Retrieved 2012-11-14, from <http://onlinelibrary.wiley.com/doi/10.1002/cmdc.200700258/abstract> doi: 10.1002/cmdc.200700258
- Pletnev, I., Erin, A., McNaught, A., Blinov, K., Tchekhovskoi, D., & Heller, S. (2012). InChIKey Collision Resistance: An Experimental Testing. *Journal of Cheminformatics*, 4(1), 39. Retrieved 2015-08-10, from <http://www.jcheminf.com/content/4/1/39/abstract> doi: 10.1186/1758-2946-4-39
- Pullan, W. (2009). Optimisation of Unweighted/Weighted Maximum Independent Sets and Minimum Vertex Covers. *Discrete Optimization*, 6(2), 214–219. Retrieved 2015-08-22, from <http://www.sciencedirect.com/science/article/pii/S1572528608000868> doi: 10.1016/j.disopt.2008.12.001
- Rahman, S. A., Bashton, M., Holliday, G. L., Schrader, R., & Thornton, J. M. (2009). Small Molecule Subgraph Detector (SMSD) toolkit. *Journal of Cheminformatics*, 1(1), 12. Retrieved 2013-07-09, from

- <http://www.jcheminf.com/content/1/1/12/abstract> doi: 10.1186/1758-2946-1-12
- R: *A Language and Environment for Statistical Computing*. (2008). Vienna, Austria: R Foundation for Statistical Computing.
- Rarey, M., Kramer, B., Lengauer, T., & Klebe, G. (1996). A Fast Flexible Docking Method using an Incremental Construction Algorithm. *Journal of Molecular Biology*, 261(3), 470–489. Retrieved 2015-08-29, from <http://www.sciencedirect.com/science/article/pii/S0022283696904775> doi: 10.1006/jmbi.1996.0477
- Ratcliff, R., & McKoon, G. (1989). Similarity Information Versus Relational Information: Differences in the Time Course of Retrieval. *Cognitive Psychology*, 21(2), 139–155. Retrieved 2013-06-12, from <http://www.sciencedirect.com/science/article/pii/0010028589900054> doi: 10.1016/0010-0285(89)90005-4
- Raymond, J. W. (2002). *Applications of Graph-Based Similarity in Cheminformatics* (PhD Thesis, University of Sheffield, Sheffield). Retrieved 2014-12-10, from <http://www.worldcat.org/title/applications-of-graph-based-similarity-in-cheminformatics/oclc/498809142>
- Raymond, J. W., Gardiner, E. J., & Willett, P. (2002a). Heuristics for Similarity Searching of Chemical Graphs Using a Maximum Common Edge Subgraph Algorithm. *Journal of Chemical Information and Computer Sciences*, 42(2), 305–316. Retrieved 2012-11-30, from <http://dx.doi.org/10.1021/ci010381f> doi: 10.1021/ci010381f
- Raymond, J. W., Gardiner, E. J., & Willett, P. (2002b). RASCAL: Calculation of Graph Similarity Using Maximum Common Edge Subgraphs. *The Computer Journal*, 45(6), 631–644. Retrieved 2012-11-22, from <http://comjnl.oxfordjournals.org/content/45/6/631.short>
- Raymond, J. W., & Willett, P. (2002a). Effectiveness of Graph-Based and Fingerprint-Based Similarity Measures for Virtual Screening of 2d Chemical Structure Databases. *Journal of Computer-Aided Molecular Design*, 16(1), 59–71. Retrieved 2012-11-19, from <http://link.springer.com/article/10.1023/A:1016387816342> doi: 10.1023/A:1016387816342
- Raymond, J. W., & Willett, P. (2002b). Maximum Common Subgraph Isomorphism Algorithms for the Matching of Chemical Structures. *Journal of Computer-Aided Molecular Design*, 16(7), 521–533.
- Richmond, N., Abrams, C., Wolohan, P., Abrahamian, E., Willett, P., & Clark, R. (2006). GALAHAD: 1. Pharmacophore Identification by Hypermolecular Alignment of Ligands in 3d. *Journal of Computer-Aided Molecular Design*, 20(9), 567–587. Retrieved 2012-10-30, from <http://www.springerlink.com/content/u880727808161321/abstract/> doi: 10.1007/s10822-006-9082-y
- Richmond, N. J., Willett, P., & Clark, R. D. (2004). Alignment of Three-Dimensional Molecules Using an Image Recognition Algorithm. *Journal of Molecular Graphics and Modelling*, 23(2), 199–209. Retrieved 2013-10-28, from <http://www.sciencedirect.com/science/article/pii/S1093326304000592> doi: 10.1016/j.jmgs.2004.04.004
- Riniker, S., & Landrum, G. A. (2013). Open-Source Platform to Benchmark Fingerprints for Ligand-

- Based Virtual Screening. *Journal of Cheminformatics*, 5(1). Retrieved 2014-03-24, from <http://www.jcheminf.com/content/5/1/26/abstract> doi: 10.1186/1758-2946-5-26
- Robertson, S. E., & Spärck Jones, K. (1976). Relevance Weighting of Search Terms. *Journal of the American Society for Information Science*, 27(3), 129–146. Retrieved 2013-09-16, from <http://onlinelibrary.wiley.com/doi/10.1002/asi.4630270302/abstract> doi: 10.1002/asi.4630270302
- Rogers, D., & Hahn, M. (2010). Extended-Connectivity Fingerprints. *Journal of Chemical Information and Modeling*, 50(5), 742–754. Retrieved 2012-11-16, from <http://pubs.acs.org/doi/abs/10.1021/ci100050t>
- Rogers, D., & Tanimoto, T. (1960). A Computer Program for Classifying Plants. *Science*, 132(3434), 1115–1118. Retrieved 2012-11-19, from <http://dx.doi.org/10.1126/science.132.3434.1115> doi: 10.1126/science.132.3434.1115
- Rohrer, S. G., & Baumann, K. (2009). Maximum Unbiased Validation (MUV) Data Sets for Virtual Screening Based on PubChem Bioactivity Data. *Journal of Chemical Information and Modeling*, 49(2), 169–184. Retrieved 2013-08-06, from <http://dx.doi.org/10.1021/ci8002649> doi: 10.1021/ci8002649
- Salton, G., & Buckley, C. (1988). Term-weighting Approaches in Automatic Text Retrieval. *Information Processing and Management*, 24(5), 513–523. Retrieved 2014-08-26, from [http://dx.doi.org/10.1016/0306-4573\(88\)90021-0](http://dx.doi.org/10.1016/0306-4573(88)90021-0) doi: 10.1016/0306-4573(88)90021-0
- Sanfeliu, A., & Fu, K.-S. (1983). A Distance Measure between Attributed Relational Graphs for Pattern Recognition. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-13(3), 353–362. Retrieved 2013-06-12, from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6313167 doi: 10.1109/TSMC.1983.6313167
- Schneider, G., Schneider, P., & Renner, S. (2006). Scaffold-Hopping: How Far Can You Jump? *QSAR & Combinatorial Science*, 25(12), 1162–1171. Retrieved 2013-07-08, from <http://onlinelibrary.wiley.com/doi/10.1002/qsar.200610091/abstract> doi: 10.1002/qsar.200610091
- Schuffenhauer, A. (2012). Computational Methods for Scaffold Hopping. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 2(6), 842–867. Retrieved 2013-07-08, from <http://onlinelibrary.wiley.com/doi/10.1002/wcms.1106/abstract> doi: 10.1002/wcms.1106
- Senger, S. (2009). Using Tversky Similarity Searches for Core Hopping: Finding the Needles in the Haystack. *Journal of Chemical Information and Modeling*, 49(6), 1514–1524. Retrieved 2013-06-21, from <http://dx.doi.org/10.1021/ci900092y> doi: 10.1021/ci900092y
- Shanmugasundaram, V., & Maggiora, G. (2001). Characterizing Property and Activity Landscapes Using an Information-Theoretic Approach. Chicago, IL, United States: American Chemical Society. Retrieved 2013-11-19, from <http://acscinf.org/docs/meetings/222nm/presentations/222nm32.pdf>
- Sheridan, R. P., & Kearsley, S. K. (2002). Why Do We Need so Many Chemical Similarity Search Methods? *Drug Discovery Today*, 7(17), 903–911. Retrieved 2013-06-21, from <http://www.sciencedirect.com/science/article/pii/S135964460202411X> doi: 10.1016/S1359-6446(02)02411-X

- Sheridan, R. P., Miller, M. D., Underwood, D. J., & Kearsley, S. K. (1996). Chemical Similarity Using Geometric Atom Pair Descriptors. *Journal of Chemical Information and Computer Sciences*, 36(1), 128–136. Retrieved 2013-09-19, from <http://dx.doi.org/10.1021/ci950275b> doi: 10.1021/ci950275b
- Simon, Z., Badilescu, I., & Racovitan, T. (1977). Mapping of Dihydrofolate-Reductase Receptor Site by Correlation with Minimal Topological (steric) Differences. *Journal of Theoretical Biology*, 66(3), 485–495. Retrieved 2013-10-08, from <http://www.sciencedirect.com/science/article/pii/0022519377902983> doi: 10.1016/0022-5193(77)90298-3
- Sirageldin, A., Selamat, A., & Ibrahim, R. (2011). Graph-Based Simulated Annealing and Support Vector Machine in Malware Detection. In *5th Malaysian Conference in Software Engineering (MySEC), 2011* (pp. 512–515). doi: 10.1109/MySEC.2011.6140720
- Spärck Jones, K. (1972). A Statistical Interpretation of Term Specificity and its Application in Retrieval. *Journal of Documentation*, 28(1), 11–21. Retrieved 2013-09-16, from <http://www.emeraldinsight.com/journals.htm?articleid=1649768&show=abstract> doi: 10.1108/eb026526
- Spoerri, A. (2008). Authority and Ranking Effects in Data Fusion. *Journal of the American Society for Information Science and Technology*, 59(3), 450–460. Retrieved 2013-11-12, from <http://onlinelibrary.wiley.com/doi/10.1002/asi.20760/abstract> doi: 10.1002/asi.20760
- Stein, S., Heller, S., & Tchekhovski, D. (2003). An Open Standard for Chemical Structure Representation - The IUPAC Chemical Identifier. In (pp. 131–143). Infonortics. Retrieved from <http://www.hellers.com/steve/resume/p157.html>
- Steinbeck, C., Han, Y., Kuhn, S., Horlacher, O., Luttmann, E., & Willighagen, E. (2003). The Chemistry Development Kit (CDK): An Open-Source Java Library for Chemo- and Bioinformatics. *Journal of Chemical Information and Computer Sciences*, 43(2), 493–500. Retrieved 2013-07-09, from <http://dx.doi.org/10.1021/ci025584y> doi: 10.1021/ci025584y
- Stiefl, N., & Zaliani, A. (2006). A Knowledge-Based Weighting Approach to Ligand-Based Virtual Screening. *Journal of Chemical Information and Modeling*, 46(2), 587–596. Retrieved 2013-02-14, from <http://dx.doi.org/10.1021/ci050324c> doi: 10.1021/ci050324c
- Svensson, F., Karlén, A., & Sköld, C. (2012). Virtual Screening Data Fusion Using Both Structure- and Ligand-Based Methods. *Journal of Chemical Information and Modeling*, 52(1), 225–232. Retrieved 2013-09-24, from <http://dx.doi.org/10.1021/ci2004835> doi: 10.1021/ci2004835
- Sylvester, J. J. (1878). On an Application of the New Atomic Theory to the Graphical Representation of the Invariants and Covariants of Binary Quantics, with Three Appendices. *American Journal of Mathematics*, 1(1), 64–104. Retrieved 2012-11-05, from <http://www.jstor.org/stable/2369436> doi: 10.2307/2369436
- Takahashi, Y., Satoh, Y., Suzuki, H., & Sasaki, S.-i. (1987). Recognition of Largest Common Structural Fragment among a Variety of Chemical Structures. *Analytical Sciences*, 3(1), 23–28. Retrieved 2014-09-18, from <http://pdf.lookchem.com/pdf/22/6cfa36a0-ef94-492a-98cc-4bd18c0ceac7.pdf>

- doi: 10.2116/analsci.3.23
- Takahashi, Y., Sukekawa, M., & Sasaki, S. (1992). Automatic Identification of Molecular Similarity Using Reduced-Graph Representation of Chemical Structure. *Journal of Chemical Information and Computer Sciences*, 32(6), 639–643. Retrieved 2014-09-17, from <http://dx.doi.org/10.1021/ci00010a009> doi: 10.1021/ci00010a009
- Teixeira, A. L., & Falcao, A. O. (2013). Noncontiguous Atom Matching Structural Similarity Function. *Journal of Chemical Information and Modeling*. Retrieved 2013-10-25, from <http://dx.doi.org/10.1021/ci400324u> doi: 10.1021/ci400324u
- Teixeira, A. L., & Falcao, A. O. (2014). Structural Similarity Based Kriging for Quantitative Structure Activity and Property Relationship Modeling. *Journal of Chemical Information and Modeling*, 54(7), 1833–1849. Retrieved 2015-08-13, from <http://dx.doi.org/10.1021/ci500110v> doi: 10.1021/ci500110v
- Tetko, I. V., Gasteiger, J., Todeschini, R., Mauri, A., Livingstone, D., Ertl, P., . . . Prokopenko, V. V. (2005). Virtual Computational Chemistry Laboratory – Design and Description. *Journal of Computer-Aided Molecular Design*, 19(6), 453–463. Retrieved 2015-08-13, from <http://link.springer.com/article/10.1007/s10822-005-8694-y> doi: 10.1007/s10822-005-8694-y
- Todeschini, R., Consonni, V., Xiang, H., Holliday, J., Buscema, M., & Willett, P. (2012). Similarity Coefficients for Binary Chemoinformatics Data: Overview and Extended Comparison Using Simulated and Real Data Sets. *Journal of Chemical Information and Modeling*, 52(11), 2884–2901. Retrieved 2013-01-06, from <http://dx.doi.org/10.1021/ci300261r> doi: 10.1021/ci300261r
- Truchon, J.-F., & Bayly, C. I. (2007). Evaluating Virtual Screening Methods: Good and Bad Metrics for the "Early Recognition" Problem. *Journal of Chemical Information and Modeling*, 47(2), 488–508. Retrieved 2013-07-11, from <http://dx.doi.org/10.1021/ci600426e> doi: 10.1021/ci600426e
- Tsai, W.-H., & Fu, K.-S. (1979). Error-Correcting Isomorphisms of Attributed Relational Graphs for Pattern Analysis. *IEEE Transactions on Systems, Man and Cybernetics*, 9(12), 757–768. doi: 10.1109/TSMC.1979.4310127
- Tsai, W.-H., & Fu, K.-S. (1983). Subgraph Error-Correcting Isomorphisms for Syntactic Pattern Recognition. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-13(1), 48–62. doi: 10.1109/TSMC.1983.6313029
- Tversky, A. (1977). Features of Similarity. *Psychological Review*, 84(4), 327–352. Retrieved 2013-06-18, from <http://homepage.psy.utexas.edu/homepage/group/loveLAB/love/classes/concepts/Tversky1977.pdf> doi: 10.1037/0033-295X.84.4.327
- Ugurlu, O. (2012). New Heuristic Algorithm for Unweighted Minimum Vertex Cover. In *Problems of Cybernetics and Informatics (PCI), 2012 IV International Conference* (pp. 1–4). Baku, Azerbaijan: IEEE. Retrieved 2015-03-14, from <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6486444> doi: 10.1109/ICPCI.2012.6486444

- Ullmann, J. R. (1976). An Algorithm for Subgraph Isomorphism. *Journal of the ACM (JACM)*, 23(1), 31–42. Retrieved 2012-10-12, from <http://dl.acm.org/citation.cfm?id=321925>
- Umeyama, S. (1988). An Eigendecomposition Approach to Weighted Graph Matching Problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5), 695–703. Retrieved 2012-11-19, from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6778
- van Berlo, R. J., de Groot, M. J., Reinders, M. J., & de Ridder, D. (2009). *Efficient Calculation of Compound Similarity Based on Maximum Common Subgraphs and Its Application to Prediction of Gene Transcript Levels* (Technical Report). Delft: Delft University of Technology.
- Varkony, T. H., Shiloach, Y., & Smith, D. H. (1979). Computer-Assisted Examination of Chemical Compounds for Structural Similarities. *Journal of Chemical Information and Computer Sciences*, 19(2), 104–111. Retrieved 2012-10-18, from <http://dx.doi.org/10.1021/ci60018a014> doi: 10.1021/ci60018a014
- Vladutz, G., & Gould, S. R. (1988). Joint Compound/Reaction Storage and Retrieval and Possibilities of a Hyperstructure-Based Solution. In *Chemical Structures: the International Language of Chemistry* (Vol. 1, pp. 371–384). Springer Verlag: Berlin. Retrieved from <http://link.springer.com/book/10.1007/978-3-642-73975-0/page/1>
- Vollmer, J. J. (1983). Wiswesser Line Notation: an Introduction. *Journal of Chemical Education*, 60(3), 192–196. Retrieved 2012-12-10, from <http://dx.doi.org/10.1021/ed060p192> doi: 10.1021/ed060p192
- Wallis, W. D., Shoubridge, P., Kraetz, M., & Ray, D. (2001). Graph Distances Using Graph Union. *Pattern Recognition Letters*, 22(6-7), 701–704. Retrieved 2012-11-19, from [http://dx.doi.org/10.1016/S0167-8655\(01\)00022-8](http://dx.doi.org/10.1016/S0167-8655(01)00022-8) doi: 10.1016/S0167-8655(01)00022-8
- Wang, Y., Eckert, H., & Bajorath, J. (2007). Apparent Asymmetry in Fingerprint Similarity Searching is a Direct Consequence of Differences in Bit Densities and Molecular Size. *ChemMedChem*, 2(7), 1037–1042. Retrieved 2013-06-20, from <http://onlinelibrary.wiley.com/doi/10.1002/cmdc.200700050/abstract> doi: 10.1002/cmdc.200700050
- Wang, Y., Ni, X., Sun, J.-T., Tong, Y., & Chen, Z. (2011). Representing Document as Dependency Graph for Document Clustering. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management* (pp. 2177–2180). Glasgow, Scotland. doi: 10.1145/2063576.2063920
- Wawer, M., Peltason, L., Weskamp, N., Teckentrup, A., & Bajorath, J. (2008). Structure-Activity Relationship Anatomy by Network-like Similarity Graphs and Local Structure-Activity Relationship Indices. *Journal of Medicinal Chemistry*, 51(19), 6075–6084. Retrieved 2013-11-26, from <http://dx.doi.org/10.1021/jm800867g> doi: 10.1021/jm800867g
- Weininger, D. (1988). SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules. *Journal Of Chemical Information And Computer Sciences*, 28(1), 31–36. Retrieved 2012-11-06, from <http://dx.doi.org/10.1021/ci00057a005> doi: 10.1021/ci00057a005

- Weininger, D., Weininger, A., & Weininger, J. (1989). SMILES .2. Algorithm For Generation Of Unique SMILES Notation. *Journal Of Chemical Information And Computer Sciences*, 29(2), 97–101.
- Wermuth, C. G., Ganellin, C. R., Lindberg, P., & Mitscher, L. A. (1998). Glossary of Terms Used in Medicinal Chemistry (IUPAC Recommendations 1998). *Pure and Applied Chemistry*, 70(5), 1129–1143. Retrieved 2012-11-14, from <http://iupac.org/publications/pac/70/5/1129/> doi: 10.1351/pac199870051129
- White, D., & Wilson, R. C. (2010). Generative Models for Chemical Structures. *Journal of Chemical Information and Modeling*, 50(7), 1257–1274. Retrieved 2015-09-28, from <http://dx.doi.org/10.1021/ci9004089> doi: 10.1021/ci9004089
- Whitney, H. (1932). Congruent Graphs and the Connectivity of Graphs. *American Journal of Mathematics*, 54(1), 150–168. Retrieved 2012-11-22, from <http://www.jstor.org/stable/2371086> doi: 10.2307/2371086
- Whittle, M., Gillet, V. J., Willett, P., Alex, A., & Loesel, J. (2004). Enhancing the Effectiveness of Virtual Screening by Fusing Nearest Neighbor Lists: A Comparison of Similarity Coefficients. *Journal of Chemical Information and Computer Sciences*, 44(5), 1840–1848. Retrieved 2013-08-05, from <http://dx.doi.org/10.1021/ci049867x> doi: 10.1021/ci049867x
- Whittle, M., Gillet, V. J., Willett, P., & Loesel, J. (2006a). Analysis of Data Fusion Methods in Virtual Screening: Similarity and Group Fusion. *Journal of Chemical Information and Modeling*, 46(6), 2206–2219. Retrieved 2013-09-20, from <http://dx.doi.org/10.1021/ci0496144> doi: 10.1021/ci0496144
- Whittle, M., Gillet, V. J., Willett, P., & Loesel, J. (2006b). Analysis of Data Fusion Methods in Virtual Screening: Theoretical Model. *Journal of Chemical Information and Modeling*, 46(6), 2193–2205. Retrieved 2013-09-20, from <http://dx.doi.org/10.1021/ci049615w> doi: 10.1021/ci049615w
- Wilkins, C. L., & Randić, M. (1980). A Graph Theoretical Approach to Structure-Property and Structure-Activity Correlations. *Theoretica chimica acta*, 58(1), 45–68. Retrieved 2015-08-12, from <http://link.springer.com/article/10.1007/BF00635723> doi: 10.1007/BF00635723
- Willett, P. (1987). *Similarity and Clustering in Chemical Information Systems*. Research Studies Press.
- Willett, P. (2009). Similarity Methods in Chemoinformatics. *Annual Review of Information Science and Technology*, 43(1), 1–117. Retrieved 2012-11-19, from <http://onlinelibrary.wiley.com/doi/10.1002/aris.2009.1440430108/abstract> doi: 10.1002/aris.2009.1440430108
- Willett, P. (2013). Combination of Similarity Rankings Using Data Fusion. *Journal of Chemical Information and Modeling*, 53(1), 1–10. Retrieved 2013-09-19, from <http://dx.doi.org/10.1021/ci300547g> doi: 10.1021/ci300547g
- Willett, P., & Winterman, V. (1986). A Comparison of Some Measures for the Determination of Inter-Molecular Structural Similarity Measures of Inter-Molecular Structural Similarity. *Quantitative Structure-Activity Relationships*, 5(1), 18–25. Retrieved 2013-07-04, from <http://onlinelibrary.wiley.com/doi/10.1002/qsar.19860050105/abstract> doi: 10.1002/qsar.19860050105

- Williams, A. J., Tkachenko, V., Golotvin, S., Kidd, R., & McCann, G. (2010). ChemSpider - Building a Foundation for the Semantic Web by Hosting a Crowd Sourced Databasing Platform for Chemistry. *Journal of Cheminformatics*, 2(Suppl 1), O16. Retrieved 2012-11-07, from <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2867124/> doi: 10.1186/1758-2946-2-S1-O16
- Wiswesser, W. J. (1954). *A Line-Formula Chemical Notation*. Crowell.
- Xue, L., & Bajorath, J. (1999). Distribution of Molecular Scaffolds and R-groups Isolated from Large Compound Databases. *Journal of Molecular Modeling*, 5(5), 97–102. Retrieved 2013-03-12, from <http://pubs.acs.org/doi/abs/10.1021/jm200452d>
- Yoshida, H., Shida, T., & Kindo, T. (2001). Asymmetric Similarity with Modified Overlap Coefficient Among Documents. In *2001 IEEE Pacific Rim Conference on Communications, Computers and signal Processing, 2001. PACRIM (Vol. 1, pp. 99–102 vol.1)*. Retrieved 2013-06-21, from <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=953532&userType=inst> doi: 10.1109/PACRIM.2001.953532
- Zarghi, A., & Arfaei, S. (2011). Selective COX-2 Inhibitors: A Review of Their Structure-Activity Relationships. *Iranian Journal of Pharmaceutical Research : IJPR*, 10(4), 655–683. Retrieved 2015-06-15, from <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3813081/>
- Zhao, X., Yuan, M., Huang, B., Ji, H., & Zhu, L. (2010). Ligand-Based Pharmacophore Model of N-Aryl and N-Heteroaryl Piperazine α 1a-Adrenoceptors Antagonists Using GALAHAD. *Journal of Molecular Graphics and Modelling*, 29(2), 126–136. Retrieved 2012-11-14, from <http://www.sciencedirect.com/science/article/pii/S1093326310000720> doi: 10.1016/j.jmgm.2010.05.002
- Zhu, Y., Qin, L., Yu, J. X., Ke, Y., & Lin, X. (2011). High Efficiency and Quality: Large Graphs Matching. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management* (pp. 1755–1764). Retrieved 2012-11-20, from <http://dl.acm.org/citation.cfm?id=2063831>
- Zhu, Y., & Yu, J. (2013). *Question on Your Work of Large Graphs Matching*. (Personal Communication)

Appendix A

Hyperstructure Construction Algorithm

Data: Hyperstructure G_1 , Molecule G_2 , Vertex Mapping between the two graphs M_v , edge mapping M_e

Result: Novel Hyperstructure H

Function createHyperstructure(M_v, M_e):

```

H = G1 ; // copy G1 to H
for edge e2 ∈ G2 do
    vq1 = e2[1] ; // Obtain atom 1 from non-hyperstructure bond
    vq2 = e2[2] ; // Obtain atom 2 from non-hyperstructure bond
    // Translate non-hyperstructure atoms to hyperstructure atoms
    vh1 = Mv[vq1] ;
    vh2 = Mv[vq2] ;
    if e2 ∉ Me then
        if vh1 = null then
            | vh1 = allocateAtom ( Mv, vq1 ) ;
        end
        if vh2 = null then
            | vh2 = allocateAtom ( Mv, vq2 ) ;
        end
        allocateBond ( Me, e2, vh1, vh2 ) ;
    end
else
    if ¬vh1 ∼ vq1 then // The two atoms vh1 and vq1 do not match
        | G1 = G1 \ vh1 ;
        | vh1 = allocateAtom(Mv, vq1) ;
    end
    if ¬vh2 ∼ vq2 then
        | G1 = G1 \ vh2 ;
        | vh2 = allocateAtom(Mv, vq2) ;
    end
end

```

Function `allocateAtom(M_v, v_q):`

```
     $v_h = \text{createAtom}(v_q)$ ; // creates new hyperstructure atom from  $G_2$  atom  
     $M_v = M_v \cup v_h$ ;  
     $G_1 = G_1 \cup v_h$ ;  
    return ( $v_h$ );
```

Function `allocateBond(M_e, e_q, v_{h1}, v_{h2}):`

```
     $v_h = \text{createBond}(e_q, v_{h1}, v_{h2})$ ; // creates new hyperstructure bond from  $G_2$  bond and  
    the two specified atoms  
     $M_e = M_e \cup e_h$ ;  
     $G_1 = G_1 \cup e_h$ ;  
    return ( $e_h$ );
```

Algorithm 4: Additional functions used in the hyperstructure construction process.

Appendix B

Diversity Statistics Bar Charts

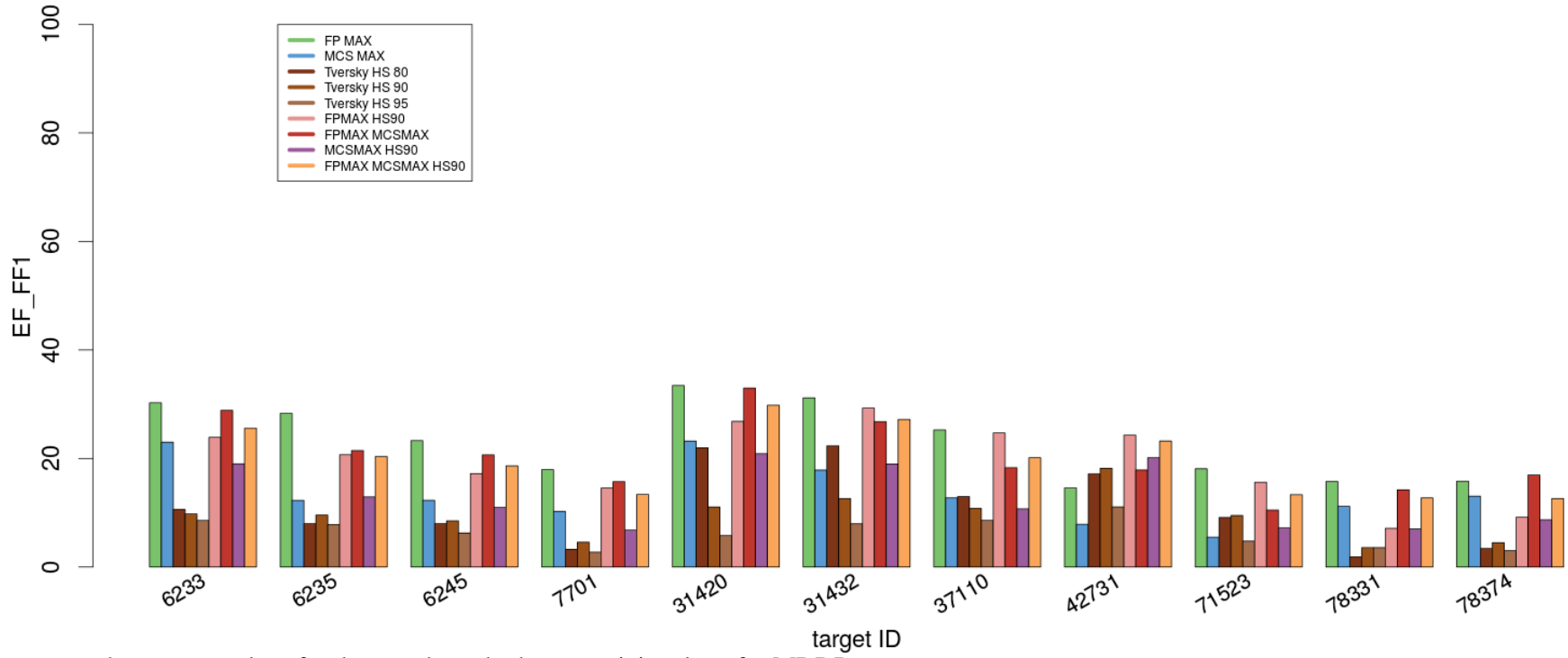


Figure B.1. $EF_{FF1\%}$ values for the search methods, per activity class, for MDDR

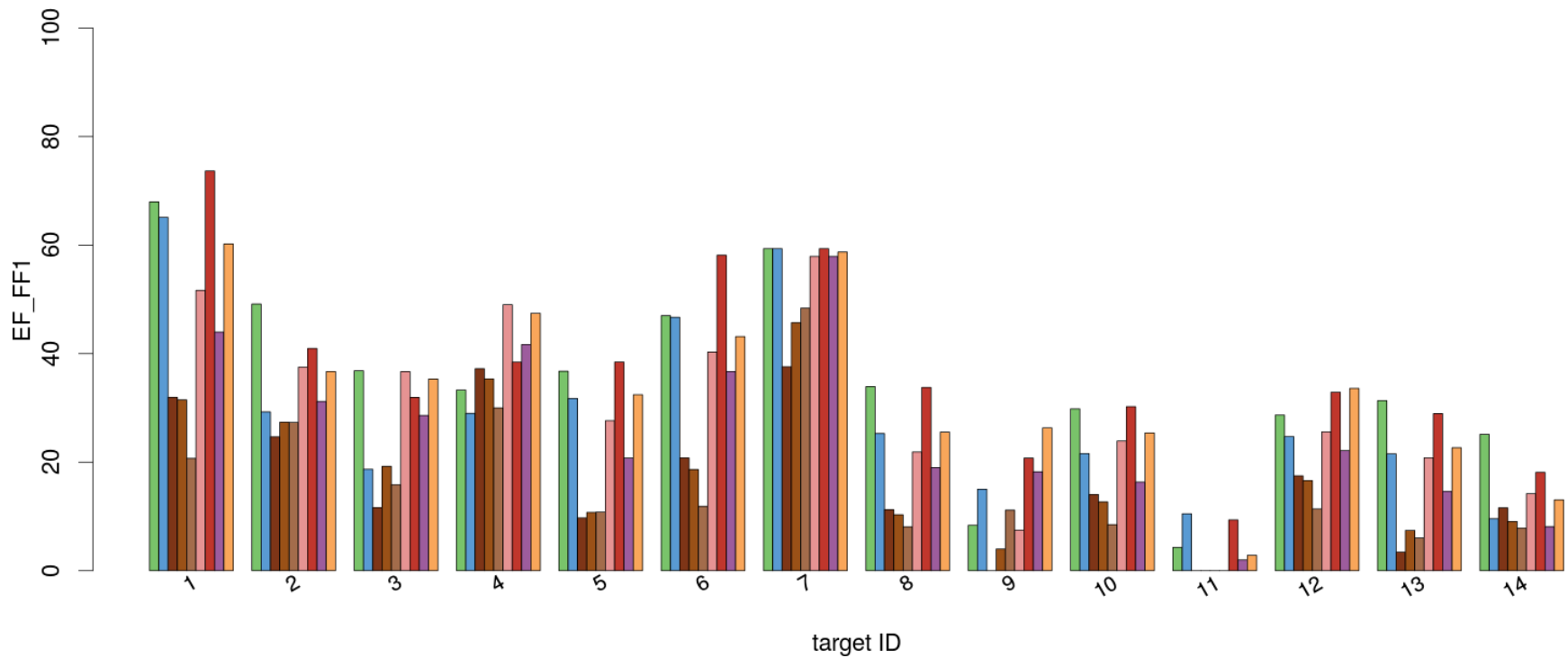


Figure B.2. $EF_{FF1\%}$ values for the search methods, per activity class, for WOMBAT

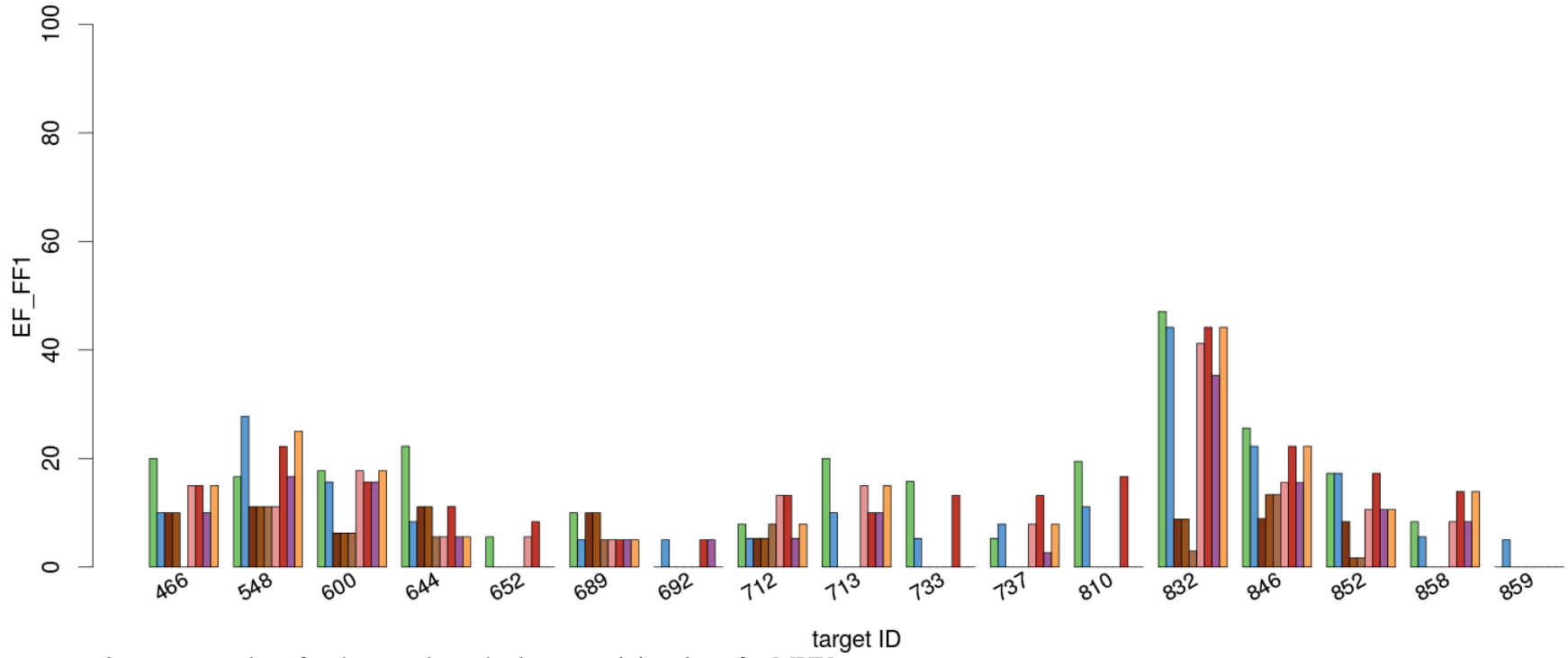


Figure B.3. $EF_{FF1\%}$ values for the search methods, per activity class, for MUV

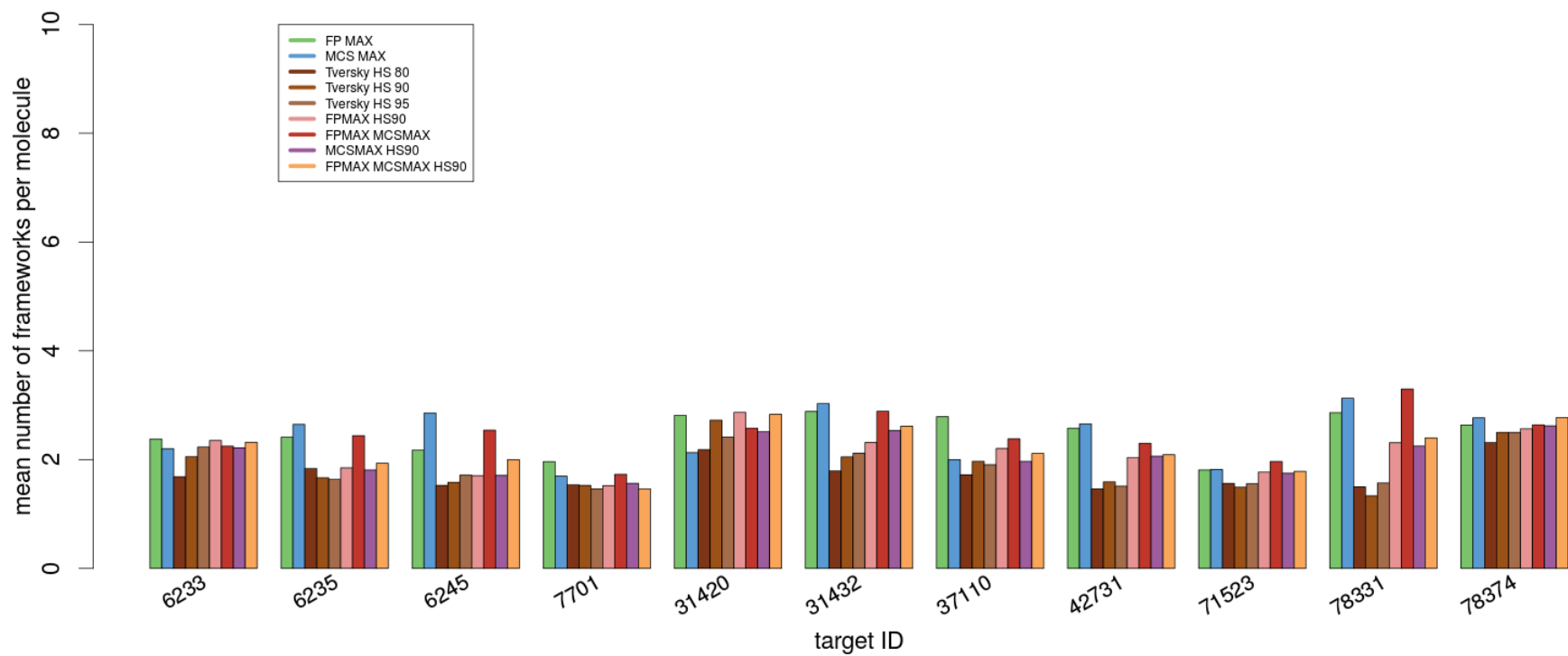


Figure B.4. \bar{x}_{F1} values for the search methods, per activity class, for MDDR

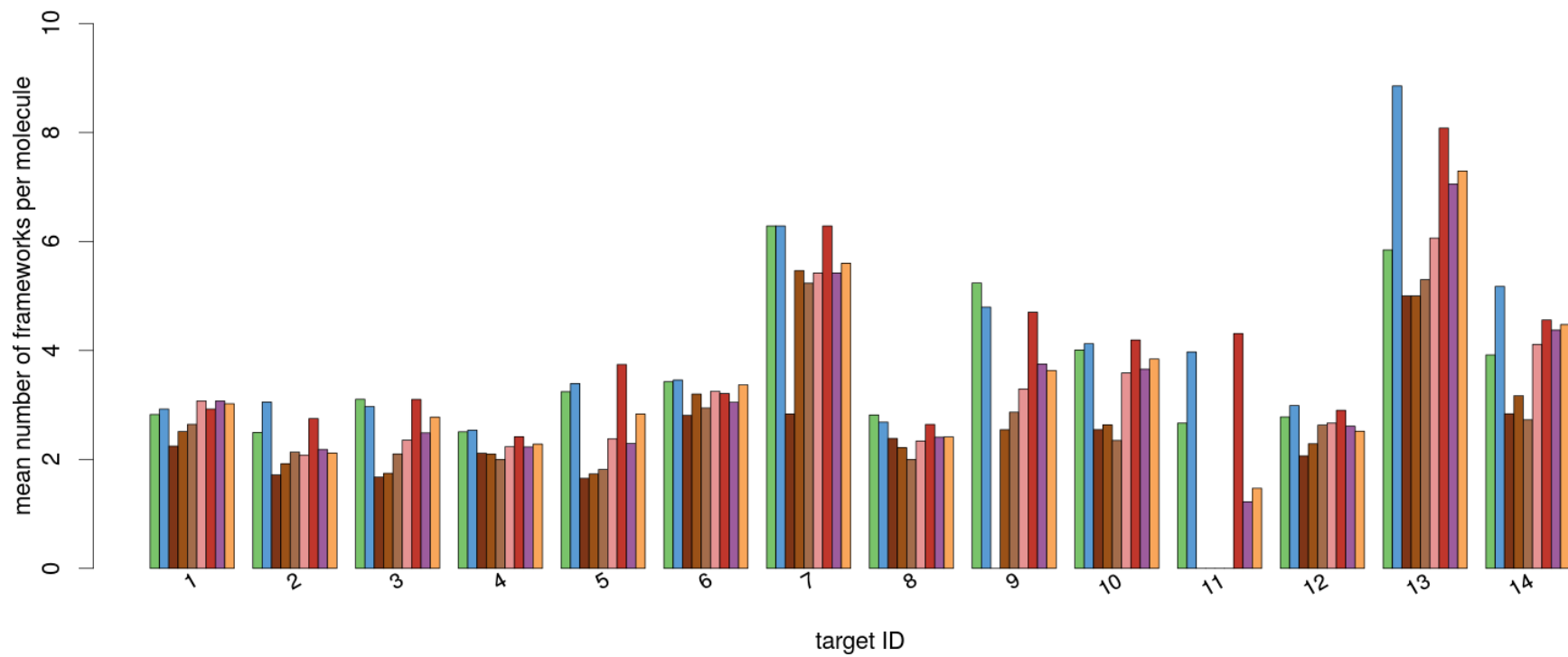


Figure B.5. \bar{x}_{F1} values for the search methods, per activity class, for WOMBAT

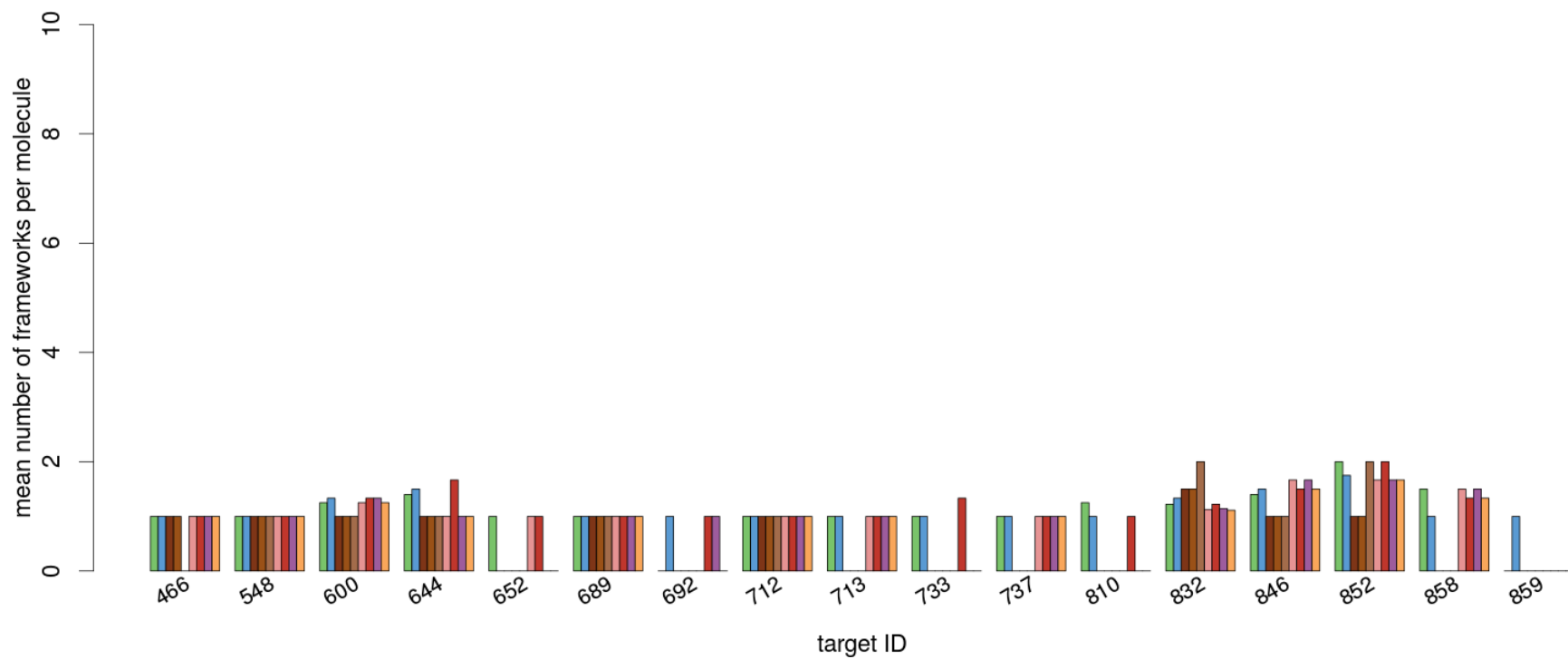
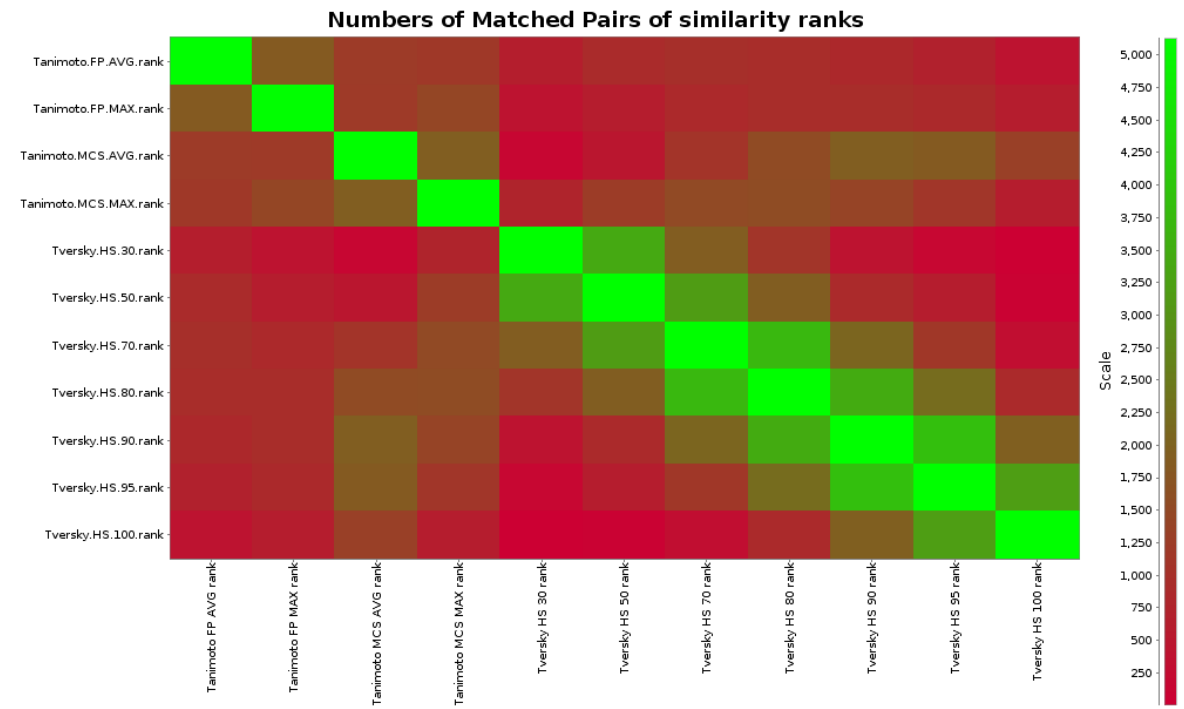


Figure B.6. \bar{x}_{F1} values for the search methods, per activity class, for MUV

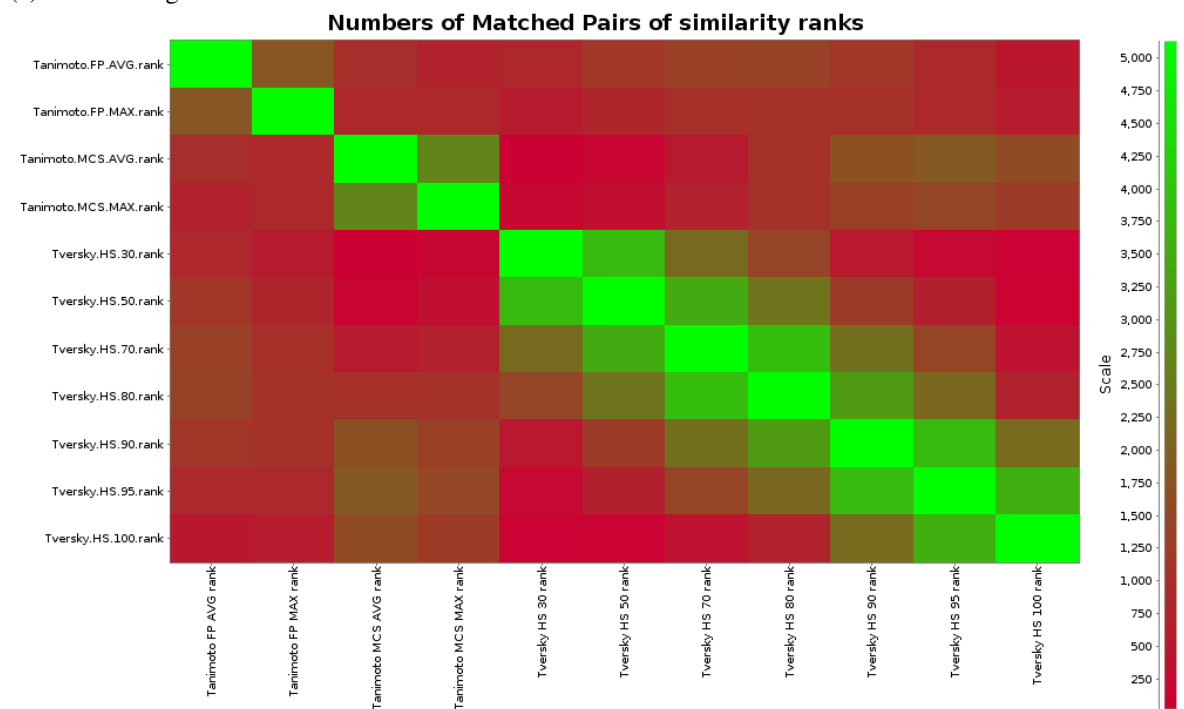
Appendix C

Similarity Search Method

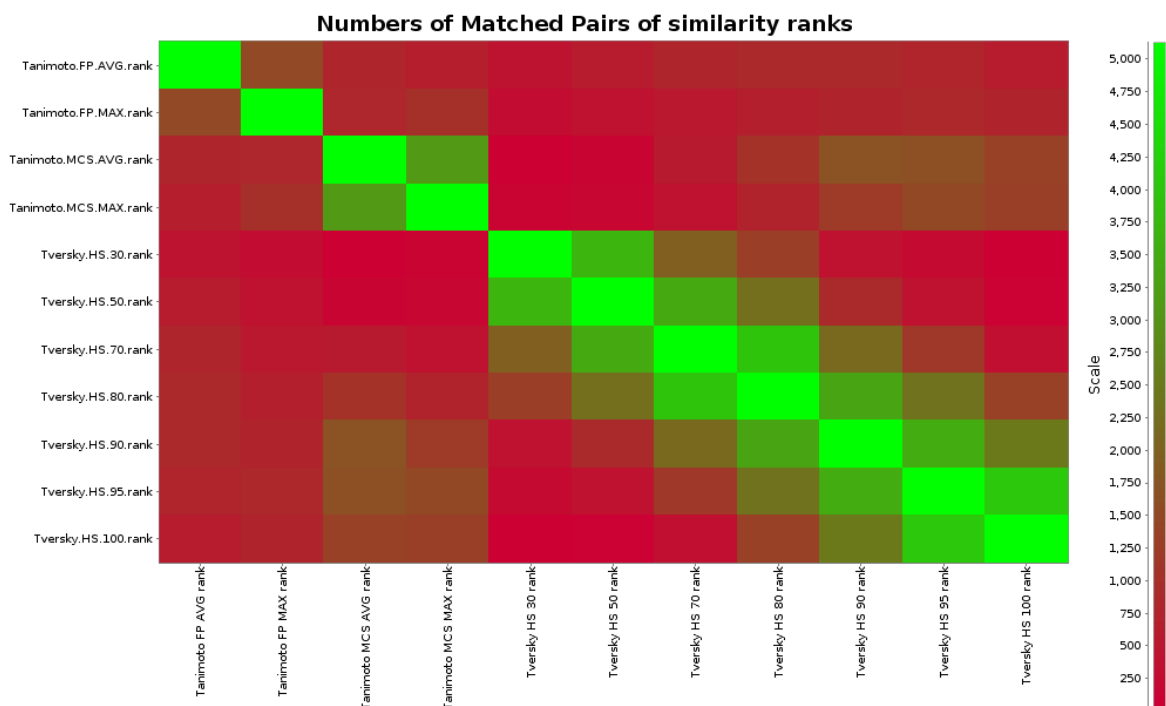
Complementarity



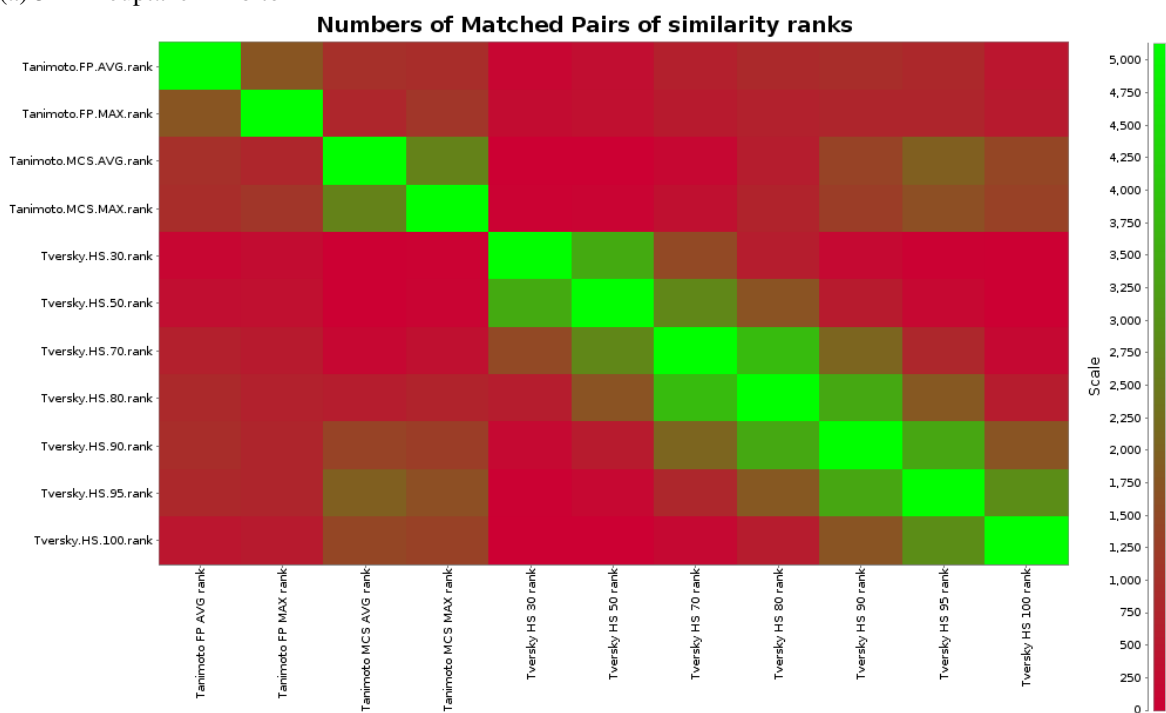
(a) 5HT3 Antagonist



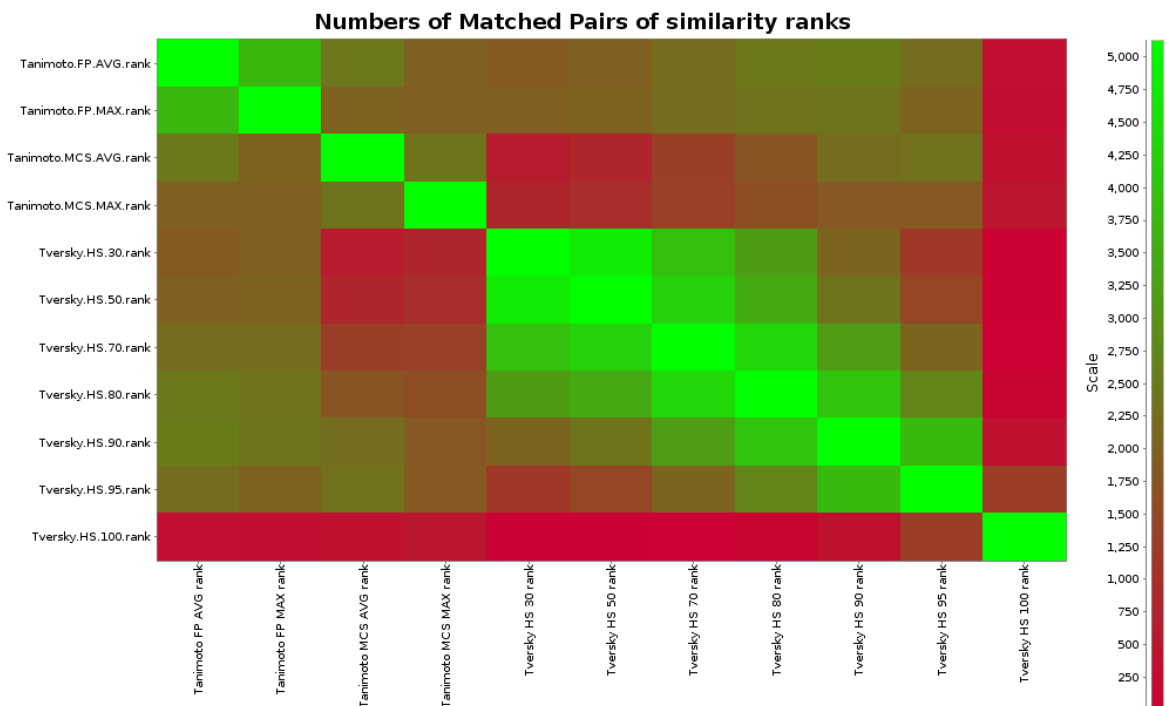
(b) 5HT1A Agonist



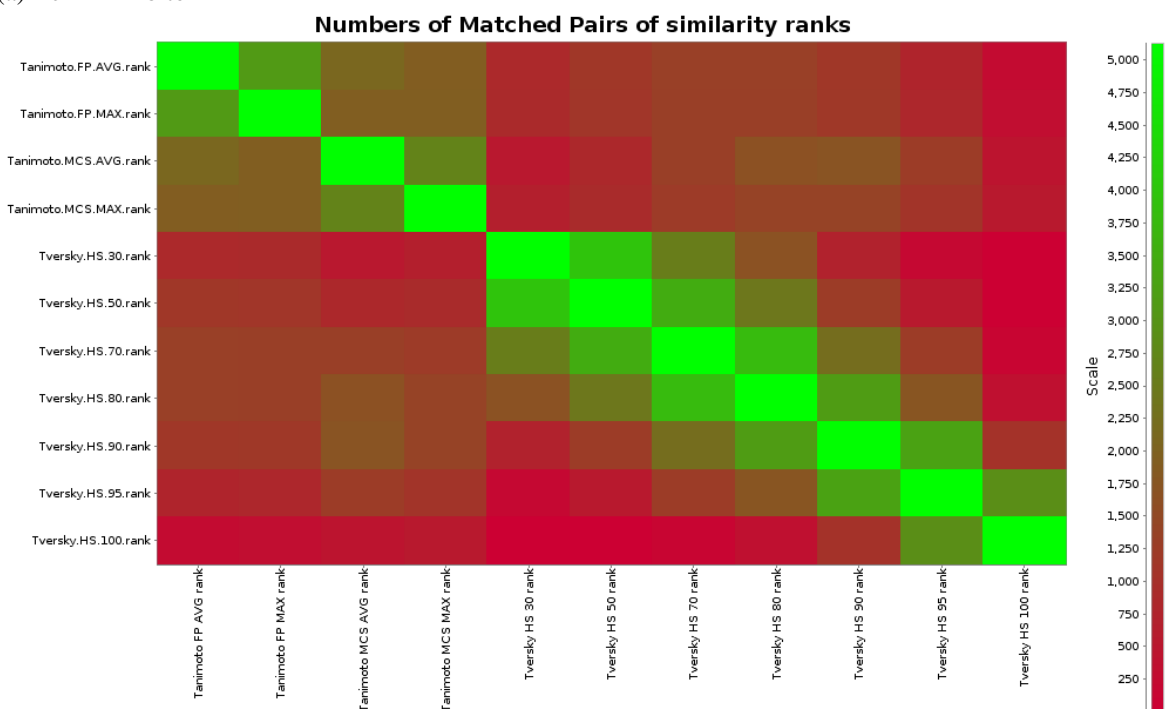
(a) 5HT Reuptake Inhibitor



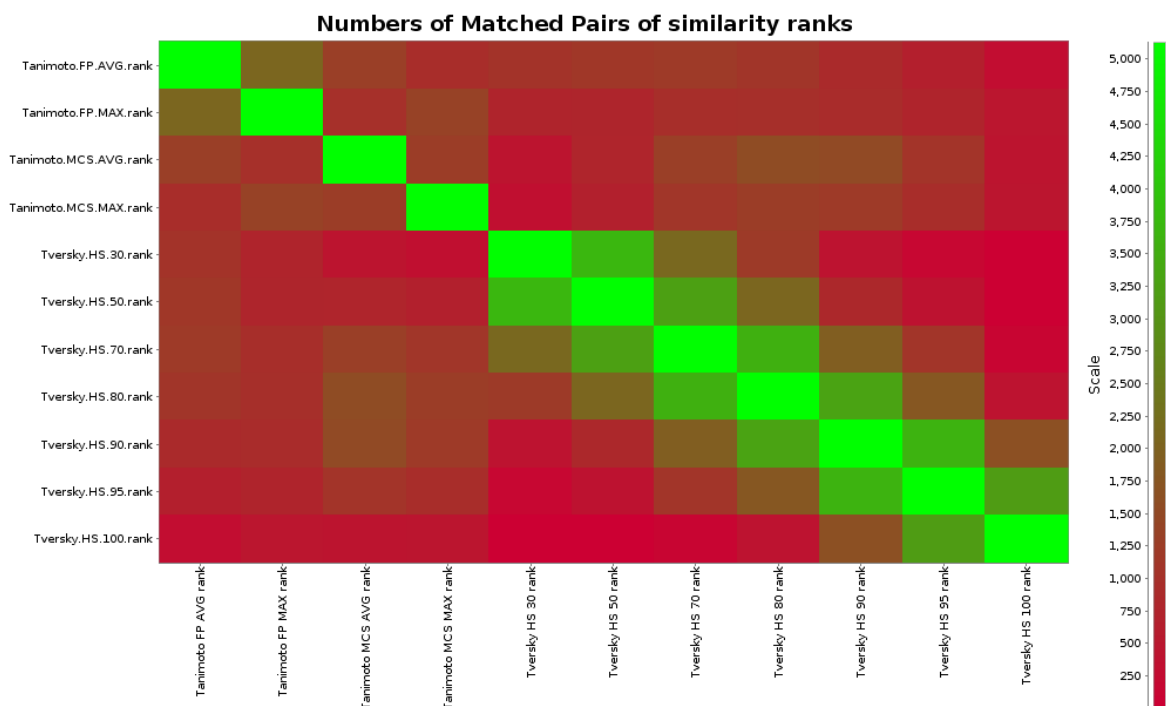
(b) D2 Antagonist



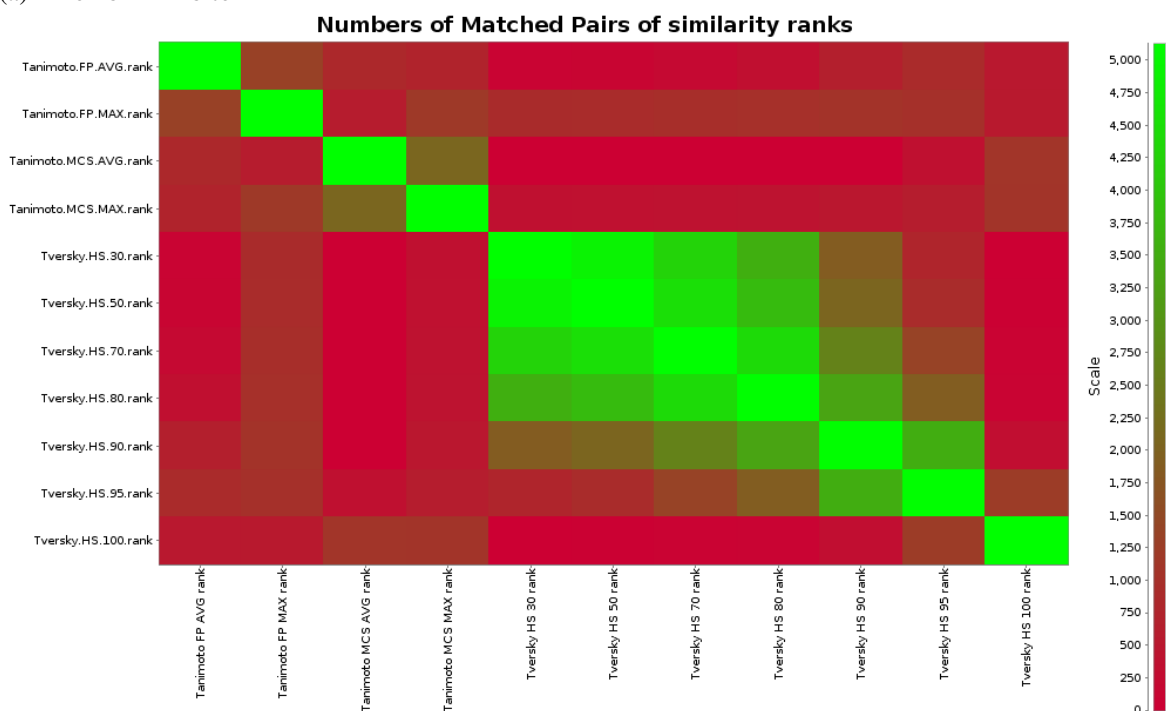
(a) Renin Inhibitor



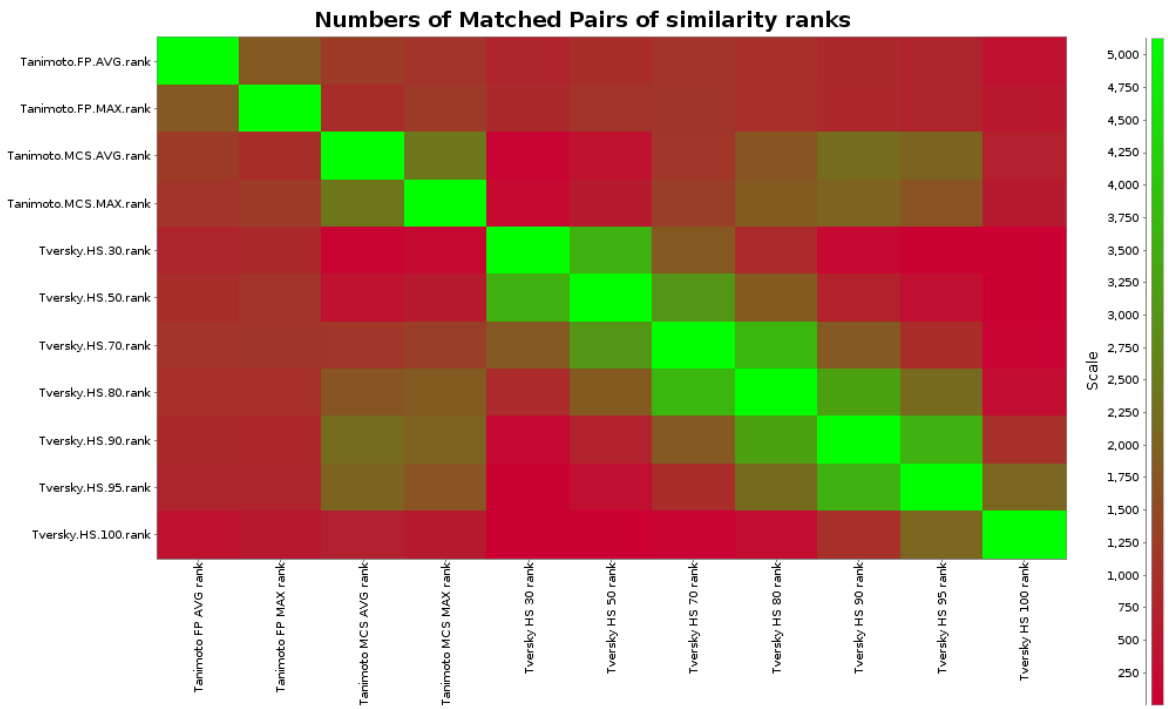
(b) Angiotensin II AT1 Antagonist



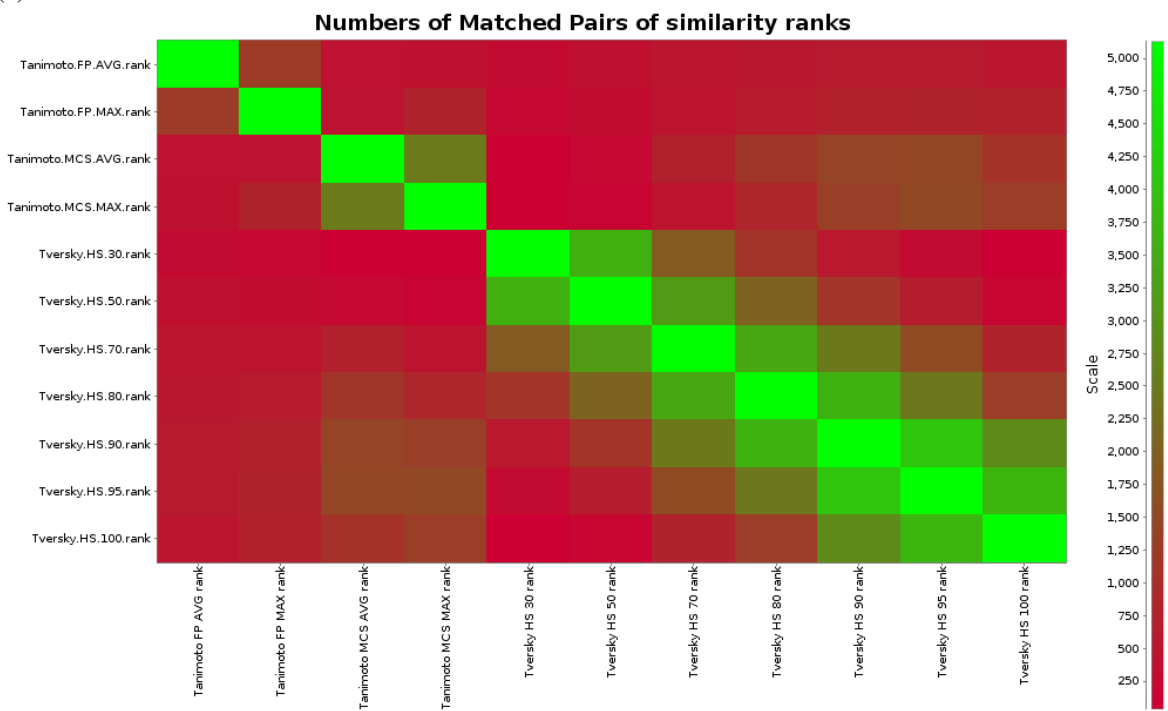
(a) Thrombin Inhibitor



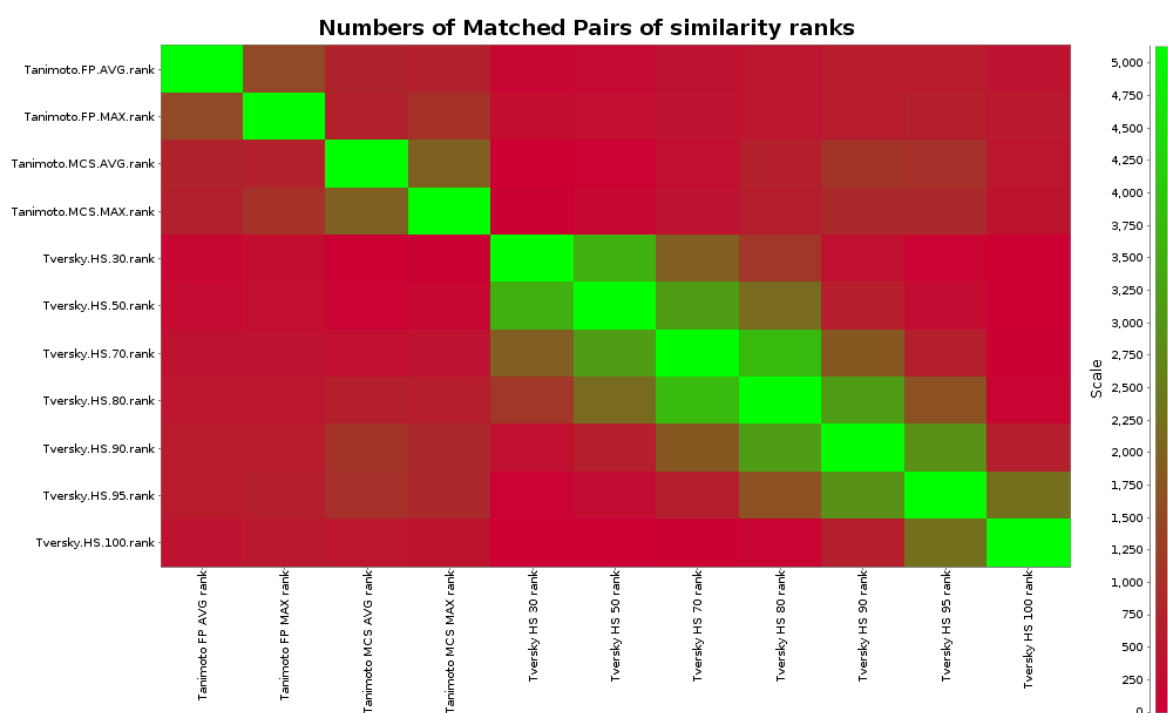
(b) Substance P Antagonist



(a) HIV-1 Protease Inhibitor



(b) Cyclooxygenase Inhibitor



(a) Protein Kinase C Inhibitor

Figure C.1. Heat maps representing the numbers of matched pairs for the ranks of the MDDR compounds tested with the methods shown. Red corresponds to low overlap, whereas green corresponds to high overlap.

Appendix D

Tabulated Results of Search Methods

class	β	EF	BEDROC	$EF_{FF1\%}$	$\bar{x}_{F1\%}$	$s_{F1\%}$	$\bar{S}_{1\%}$
5HT3 Antagonist	0.90	7.116	0.257	8.971	1.708	1.220	0.204
5HT1A Agonist	0.80	7.173	0.282	13.552	1.689	2.013	0.223
5HT Reuptake Inhibitor	0.95	7.794	0.259	8.326	2.045	1.676	0.170
D2 Antagonist	0.90	5.143	0.195	12.413	1.158	0.370	0.228
Renin Inhibitor	0.90	14.518	0.540	23.791	1.932	2.962	0.347
Angiotensin II AT1 Antagonist	0.70	8.832	0.325	13.268	1.654	1.195	0.292
Thrombin Inhibitor	0.80	8.953	0.349	17.965	1.645	1.755	0.237
Substance P Antagonist	0.95	6.537	0.249	8.165	1.609	1.517	0.192
HIV-1 Protease Inhibitor	0.50	8.838	0.317	12.127	1.474	0.977	0.218
Cyclooxygenase Inhibitor	0.95	3.898	0.134	4.036	2.100	1.447	0.135
Protein Kinase C Inhibitor	0.95	3.160	0.120	2.281	2.273	1.794	0.177

Table D.1

Hyperstructure similarity search results for the MDDR dataset, where for each class, the β of the Tversky coefficient is reported which yields the greatest BEDROC score.

class	EF	BEDROC	$EF_{FF1\%}$	$\bar{x}_{F1\%}$	$s_{F1\%}$	$\bar{S}_{1\%}$
5HT3 Antagonist	9.811	0.424	23.007	2.182	2.340	0.171
5HT1A Agonist	6.659	0.258	8.731	2.096	1.752	0.140
5HT Reuptake Inhibitor	9.284	0.370	13.400	3.129	3.810	0.139
D2 Antagonist	7.325	0.294	16.790	1.617	1.074	0.185
Renin Inhibitor	12.732	0.521	15.547	2.898	5.016	0.290
Angiotensin II AT1 Antagonist	13.848	0.541	21.016	2.527	3.701	0.309
Thrombin Inhibitor	9.231	0.381	16.752	2.136	2.991	0.203
Substance P Antagonist	5.906	0.269	7.619	3.197	6.177	0.195
HIV-1 Protease Inhibitor	5.514	0.231	10.553	1.759	1.261	0.194
Cyclooxygenase Inhibitor	5.080	0.222	10.487	2.714	3.073	0.128
Protein Kinase C Inhibitor	9.977	0.440	16.198	3.705	4.338	0.186

Table D.2
MCS MAX similarity search results for the MDDR dataset.

class	EF	BEDROC	$EF_{FF1\%}$	$\bar{x}_{F1\%}$	$s_{F1\%}$	$\bar{S}_{1\%}$
5HT3 Antagonist	13.854	0.592	30.264	2.364	3.072	0.170
5HT1A Agonist	9.523	0.384	16.293	2.216	2.714	0.189
5HT Reuptake Inhibitor	9.456	0.393	19.662	2.255	2.326	0.139
D2 Antagonist	9.974	0.404	25.821	1.700	1.418	0.193
Renin Inhibitor	19.750	0.816	31.437	2.682	5.610	0.313
Angiotensin II AT1 Antagonist	19.657	0.808	32.869	2.736	4.449	0.341
Thrombin Inhibitor	12.383	0.495	21.294	2.189	3.151	0.217
Substance P Antagonist	6.553	0.288	13.504	2.208	2.802	0.168
HIV-1 Protease Inhibitor	15.243	0.614	32.193	1.856	2.210	0.227
Cyclooxygenase Inhibitor	6.166	0.267	13.470	2.644	2.756	0.128
Protein Kinase C Inhibitor	9.932	0.385	19.689	2.417	2.751	0.165

Table D.3
FP MAX similarity search results for MDDR.

class	β	EF	BEDROC	$EF_{FF1\%}$	$\bar{x}_{F1\%}$	$s_{F1\%}$	$\bar{S}_{1\%}$
5HT3 Antagonist (rat)	0.90	18.922	0.721	39.904	2.510	3.163	0.322
5HT1A Agonist (rat)	0.90	7.708	0.298	12.987	2.439	2.627	0.187
D2 Antagonist (rat)	0.95	8.927	0.321	14.097	1.872	2.682	0.234
Renin Inhibitor (human)	0.95	4.769	0.156	3.837	1.500	0.972	0.282
Angiotensin II AT1 Antagonist (rat)	0.95	5.225	0.172	3.619	2.000	1.483	0.272
Thrombin Inhibitor (human)	0.95	12.286	0.506	25.969	3.103	3.895	0.223
Substance P Antagonist (human)	0.95	16.667	0.762	49.155	5.222	9.789	0.382
HIV-1 Protease Inhibitor (human)	0.90	5.154	0.212	15.962	1.979	2.382	0.221
Cyclooxygenase Inhibitor (human)	0.95	9.124	0.260	1.702	2.667	3.172	0.285
Protein Kinase C Inhibitor (rat)	0.70	10.168	0.342	14.945	1.932	2.102	0.200
Acetylcholine esterase inhibitor (human)	1.00	3.156	0.110	1.053	2.000	1.414	0.195
Factor Xa inhibitor (human)	0.90	9.620	0.373	17.404	2.318	2.121	0.200
Matrix metalloprotease inhibitor (human)	0.95	6.366	0.246	6.310	6.842	19.791	0.252
Phosphodiesterase inhibitor (human)	0.90	7.938	0.285	10.433	2.526	3.516	0.208

Table D.4

Hyperstructure similarity search results for the WOMBAT dataset.

class	EF	BEDROC	$EF_{FF1\%}$	$\bar{x}_{F1\%}$	$s_{F1\%}$	$\bar{S}_{1\%}$
5HT3 Antagonist (rat)	19.914	0.935	65.869	2.890	5.390	0.324
5HT1A Agonist (rat)	10.426	0.433	8.273	5.091	6.049	0.153
D2 Antagonist (rat)	8.587	0.341	10.525	3.531	6.347	0.245
Renin Inhibitor (human)	13.236	0.571	30.768	2.875	4.229	0.222
Angiotensin II AT1 Antagonist (rat)	12.163	0.533	27.676	3.810	6.068	0.237
Thrombin Inhibitor (human)	17.238	0.735	43.144	3.357	4.642	0.186
Substance P Antagonist (human)	20.000	0.986	59.296	6.286	11.051	0.363
HIV-1 Protease Inhibitor (human)	8.771	0.397	24.078	2.704	4.574	0.159
Cyclooxygenase Inhibitor (human)	13.212	0.561	15.225	5.511	7.531	0.184
Protein Kinase C Inhibitor (rat)	9.976	0.450	21.086	3.700	6.421	0.189
Acetylcholine esterase inhibitor (human)	8.133	0.294	14.670	2.771	3.657	0.183
Factor Xa inhibitor (human)	13.889	0.572	31.701	2.817	3.687	0.217
Matrix metalloprotease inhibitor (human)	9.424	0.455	10.355	13.577	50.458	0.231
Phosphodiesterase inhibitor (human)	9.278	0.358	14.130	3.442	4.692	0.182

Table D.5

MCS MAX similarity search results for the WOMBAT dataset.

class	EF	BEDROC	$EF_{FF1\%}$	$\bar{x}_{F1\%}$	$s_{F1\%}$	$\bar{S}_{1\%}$
5HT3 Antagonist (rat)	20.000	0.923	65.768	2.816	5.325	0.325
5HT1A Agonist (rat)	14.726	0.640	35.489	2.842	3.942	0.151
D2 Antagonist (rat)	13.023	0.510	17.035	3.708	6.276	0.232
Renin Inhibitor (human)	13.771	0.524	18.711	3.191	4.739	0.235
Angiotensin II AT1 Antagonist (rat)	19.157	0.722	33.080	3.844	6.416	0.243
Thrombin Inhibitor (human)	19.333	0.901	62.191	3.368	5.321	0.188
Substance P Antagonist (human)	20.000	0.991	59.296	6.286	11.051	0.363
HIV-1 Protease Inhibitor (human)	15.119	0.671	41.933	2.833	3.706	0.159
Cyclooxygenase Inhibitor (human)	11.569	0.388	4.482	5.095	5.770	0.223
Protein Kinase C Inhibitor (rat)	15.000	0.635	31.943	3.560	6.332	0.203
Acetylcholine esterase inhibitor (human)	6.222	0.218	7.528	2.419	2.446	0.248
Factor Xa inhibitor (human)	16.287	0.639	30.543	2.686	3.412	0.236
Matrix metalloprotease inhibitor (human)	10.325	0.462	17.109	8.590	29.306	0.205
Phosphodiesterase inhibitor (human)	13.711	0.537	25.820	2.931	5.198	0.187

Table D.6

FP MAX similarity search results for WOMBAT.

class	β	EF	BEDROC	$EF_{FF1\%}$	$\bar{x}_{F1\%}$	$s_{F1\%}$	$\bar{S}_{1\%}$
sphingosine-1-phosphate 1 receptor potentiators	0.95	3.000	0.073	5.000	1.000	0.000	1.000
protein kinase A inhibitors	1.00	3.000	0.059	0.000	0.000	0.000	0.000
steroidogenic factor 1 inhibitors	0.90	4.000	0.116	6.250	1.000	0.000	1.000
rho kinase 2 inhibitors	1.00	3.000	0.125	8.347	1.000	0.000	0.547
HIV reverse transcriptase RNase	0.95	2.000	0.047	0.000	0.000	0.000	0.000
ephrin type-A receptor 4 antagonist inhibitors	0.90	3.000	0.092	0.000	0.000	0.000	0.000
steroidogenic factor 1 activators	0.95	1.000	0.028	0.000	0.000	0.000	0.000
heat shock protein 90 kDa alpha inhibitors	0.95	3.000	0.132	10.526	1.000	0.000	0.611
estrogen receptor-alpha coactivator binding inhibitors	0.30	1.000	0.022	0.000	0.000	0.000	0.000
estrogen receptor-beta coactivator binding inhibitors	0.50	2.000	0.033	0.000	0.000	0.000	0.000
estrogen receptor-alpha coactivator binding potentiators	0.70	5.000	0.099	0.000	0.000	0.000	0.000
focal adhesion kinase inhibitors	0.30	5.000	0.182	11.111	1.000	0.000	0.582
Cathepsin G	0.90	11.000	0.394	20.603	1.250	0.500	0.385
factor XIa	0.70	6.000	0.274	13.745	1.667	1.155	0.424
factor XIIa	1.00	4.000	0.128	1.457	1.000	0.000	1.000
dopamine receptor D1 allosteric modulators	0.95	2.000	0.122	11.111	1.000	0.000	0.565
muscarinic receptor M1 allosteric modulators	0.30	4.000	0.164	15.000	1.000	0.000	0.441

Table D.7

Hyperstructure similarity search results for the MUV dataset. Due to the low number of actives in the MUV sets, no diversity statistics have been reported.

class	EF	BEDROC	$EF_{FF1\%}$	$\bar{x}_{F1\%}$	$s_{F1\%}$	$\bar{S}_{1\%}$
sphingosine-1-phosphate 1 receptor potentiators	3.000	0.124	5.000	1.000	0.000	1.000
protein kinase A inhibitors	8.000	0.233	2.956	1.000	0.000	1.000
steroidogenic factor 1 inhibitors	6.000	0.204	9.391	1.500	0.707	0.480
rho kinase 2 inhibitors	3.000	0.126	2.792	2.000	0.000	0.764
HIV reverse transcriptase RNase	2.000	0.080	5.263	1.000	0.000	1.000
ephrin type-A receptor 4 antagonist inhibitors	2.000	0.096	5.000	1.000	0.000	1.000
steroidogenic factor 1 activators	3.000	0.090	5.000	1.000	0.000	1.000
heat shock protein 90 kDa alpha inhibitors	3.000	0.122	5.263	1.000	0.000	1.000
estrogen receptor-alpha coactivator binding inhibitors	2.000	0.083	5.000	1.000	0.000	1.000
estrogen receptor-beta coactivator binding inhibitors	8.000	0.233	10.000	1.000	0.000	0.548
estrogen receptor-alpha coactivator binding potentiators	5.000	0.165	10.000	1.000	0.000	0.565
focal adhesion kinase inhibitors	3.000	0.106	11.111	1.000	0.000	0.545
Cathepsin G	12.000	0.538	41.206	1.250	0.463	0.283
factor XIa	9.000	0.441	37.275	1.286	0.756	0.283
factor XIIa	3.000	0.124	7.143	1.000	0.000	1.000
dopamine receptor D1 allosteric modulators	5.000	0.174	11.111	1.000	0.000	0.560
muscarinic receptor M1 allosteric modulators	4.000	0.117	5.000	1.000	0.000	1.000

Table D.8

MCS MAX similarity search results for MUV.

class	EF	BEDROC	$EF_{FF1\%}$	$\bar{x}_{F1\%}$	$s_{F1\%}$	$\bar{S}_{1\%}$
sphingosine-1-phosphate 1 receptor potentiators	4.000	0.160	10.000	1.000	0.000	0.560
protein kinase A inhibitors	7.000	0.302	11.794	1.333	0.577	0.579
steroidogenic factor 1 inhibitors	7.000	0.265	12.531	1.333	0.577	0.367
rho kinase 2 inhibitors	8.000	0.187	8.347	1.000	0.000	0.539
HIV reverse transcriptase RNase	2.000	0.100	5.263	1.000	0.000	1.000
ephrin type-A receptor 4 antagonist inhibitors	6.000	0.252	20.000	1.000	0.000	0.345
steroidogenic factor 1 activators	3.000	0.128	15.000	1.000	0.000	0.428
heat shock protein 90 kDa alpha inhibitors	5.000	0.168	7.908	1.000	0.000	0.546
estrogen receptor-alpha coactivator binding inhibitors	2.000	0.114	10.000	1.000	0.000	0.555
estrogen receptor-beta coactivator binding inhibitors	3.000	0.139	15.000	1.000	0.000	0.452
estrogen receptor-alpha coactivator binding potentiators	3.000	0.138	10.000	1.000	0.000	0.565
focal adhesion kinase inhibitors	7.000	0.271	19.458	1.250	0.500	0.328
Cathepsin G	14.000	0.579	47.088	1.222	0.441	0.259
factor XIa	13.000	0.575	40.231	1.250	0.707	0.268
factor XIIa	12.000	0.422	8.600	3.000	2.828	0.571
dopamine receptor D1 allosteric modulators	5.000	0.223	13.903	1.333	0.577	0.395
muscarinic receptor M1 allosteric modulators	2.000	0.059	5.000	1.000	0.000	1.000

Table D.9

FP MAX similarity search results for MUV.

Appendix E

Summary performances of Search

Methods

method	BEDROC	EF	BEDROC	EF	BEDROC	EF
FP MAX	0.495	12.045	0.626	14.875	0.240	6.059
FPMAX HS90	0.412	10.528	0.499	12.566	0.162	4.412
MCSMAX HS90	0.330	8.711	0.464	11.567	0.137	3.588
FPMAX MCSMAX	0.485	11.877	0.656	15.042	0.216	5.824
FPMAX MCSMAX HS90	0.443	11.336	0.583	14.238	0.178	5.000
p_FPMAX HS90	0.042	0.320	0.000	0.005	0.000	0.003
p_MCSMAX HS90	0.003	0.019	0.002	0.002	0.000	0.003
p_FPMAX MCSMAX	0.898	1.000	0.358	0.969	0.171	0.549
p_FPMAX MCSMAX HS90	0.278	0.700	0.258	0.364	0.002	0.055
	(a) <i>MDDR</i>		(b) <i>WOMBAT</i>		(c) <i>MUV</i>	

Table E.1

General comparisons between virtual screening statistics, comparing the data fusion methods to FP MAX, for the three datasets mentioned. The first set of rows quote the mean statistic scores obtained for the activity classes a method is tested on. The other rows prefixed with p_ show the p-value obtained from the paired Wilcoxon signed-rank test of the method, compared with FP MAX.

method	$EF_{FF1\%}$	$\bar{x}_{F1\%}$	$\bar{S}_{1\%}$
FP MAX	23.318	2.297	0.205
FPMAX HS90	18.163	2.067	0.210
MCSMAX HS90	12.835	1.999	0.210
FPMAX MCSMAX	21.535	2.434	0.195
FPMAX MCSMAX HS90	18.507	2.201	0.201
p_FPMAX HS90	0.024	0.001	0.465
p_MCSMAX HS90	0.001	0.001	0.594
p_FPMAX MCSMAX	0.175	0.413	0.068
p_FPMAX MCSMAX HS90	0.005	0.041	0.760

(a) *MDDR*

method	$EF_{FF1\%}$	$\bar{x}_{F1\%}$	$\bar{S}_{1\%}$
FP MAX	32.209	3.869	0.228
FPMAX HS90	25.004	3.049	0.239
MCSMAX HS90	23.564	3.382	0.252
FPMAX MCSMAX	32.315	4.240	0.220
FPMAX MCSMAX HS90	29.163	3.664	0.239
p_FPMAX HS90	0.002	0.011	0.124
p_MCSMAX HS90	0.013	0.025	0.096
p_FPMAX MCSMAX	0.944	0.069	0.053
p_FPMAX MCSMAX HS90	0.296	0.326	0.245

(b) *WOMBAT*

method	$EF_{FF1\%}$	$\bar{x}_{F1\%}$	$\bar{S}_{1\%}$
FP MAX	15.301	1.219	0.515
FPMAX HS90	9.019	0.776	0.475
MCSMAX HS90	8.267	0.778	0.496
FPMAX MCSMAX	13.330	1.130	0.628
FPMAX MCSMAX HS90	10.285	0.810	0.479
p_FPMAX HS90	0.001	0.006	1.000
p_MCSMAX HS90	0.001	0.009	0.705
p_FPMAX MCSMAX	0.230	0.675	0.068
p_FPMAX MCSMAX HS90	0.007	0.008	0.856

(c) *MUV*

Table E.2

General comparisons between diversity results, comparing the data fusion methods to FP MAX. The first set of rows quote the mean statistic scores obtained for the activity classes a method is tested on. The other rows prefixed with p_ show the p-value obtained from the paired Wilcoxon signed-rank test of the method, compared with FP MAX.

Appendix F

Mean Ranks of Methods

method	BEDROC	method	BEDROC
Tversky HS 30	1.545	Tversky HS 30	1.536
Tversky HS 50	2.636	Tversky HS 50	2.964
Tversky HS 100	2.727	Tversky HS 100	3.000
Tversky HS 70	4.636	Tversky HS 70	4.429
Tversky HS 95	5.455	Tversky HS 80	5.857
Tversky HS 80	6.182	Tversky HS 95	6.179
MCS AVG	6.727	Tversky HS 90	6.429
Tversky HS 90	7.091	MCS AVG	7.214
MCSMAX HS90	9.182	MCSMAX HS90	9.143
MCS MAX	10.091	FPMAX HS90	9.964
FPMAX HS90	11.591	MCS MAX	11.714
FPMAX MCSMAX HS90	12.227	FPMAX MCSMAX HS90	11.786
FP AVG	12.773	FP MAX	12.714
FPMAX MCSMAX	13.273	FP AVG	13.000
FP MAX	13.864	FPMAX MCSMAX	14.071
W	0.870	W	0.835
p	$1.124 \cdot 10^{-21}$	p	$1.247 \cdot 10^{-27}$

(a) *MDDR*(b) *WOMBAT*

method	BEDROC
Tversky HS 95	4.471
Tversky HS 100	4.735
Tversky HS 30	5.118
Tversky HS 50	5.412
Tversky HS 90	5.412
MCS AVG	5.765
Tversky HS 70	6.500
Tversky HS 80	7.324
MCSMAX HS90	8.324
FPMAX HS90	9.588
FPMAX MCSMAX HS90	9.735
FP AVG	10.265
MCS MAX	11.500
FPMAX MCSMAX	12.912
FP MAX	12.941
W	0.455
p	$1.218 \cdot 10^{-16}$

(c) *MUV*

Table F.1

Mean rankings of the *BEDROC* scores (a lower rank value indicating worse performance) of the methods, sorted in ascending order. Values for Kendall's *W* and the associated *p* value are given for each dataset, where Kendall's *W* quantifies the agreement in rankings between the activity classes of the dataset for a search method.

method	$EF_{FF1\%}$	method	$EF_{FF1\%}$
Tversky HS 100	2.182	Tversky HS 100	2.071
Tversky HS 30	2.227	Tversky HS 30	2.321
Tversky HS 50	3.818	Tversky HS 50	3.679
Tversky HS 95	4.091	Tversky HS 95	5.429
Tversky HS 70	5.955	Tversky HS 70	5.500
MCS AVG	6.727	Tversky HS 80	6.286
Tversky HS 90	6.909	Tversky HS 90	6.500
Tversky HS 80	7.091	MCS AVG	6.786
MCSMAX HS90	8.455	MCSMAX HS90	9.107
MCS MAX	8.636	MCS MAX	10.357
FPMAX HS90	11.909	FPMAX HS90	10.393
FPMAX MCSMAX HS90	12.273	FPMAX MCSMAX HS90	11.929
FPMAX MCSMAX	12.636	FP MAX	12.857
FP AVG	13.182	FPMAX MCSMAX	13.286
FP MAX	13.909	FP AVG	13.500
W	0.804	W	0.776
p	$1.087 \cdot 10^{-19}$	p	$2.859 \cdot 10^{-25}$

(a) *MDDR* (b) *WOMBAT*

method	$EF_{FF1\%}$
Tversky HS 100	4.647
Tversky HS 95	5.118
Tversky HS 30	5.882
Tversky HS 90	5.941
MCS AVG	6.059
Tversky HS 80	6.118
Tversky HS 70	6.471
Tversky HS 50	6.647
MCSMAX HS90	7.971
FPMAX HS90	9.441
FPMAX MCSMAX HS90	9.735
MCS MAX	10.500
FP AVG	10.971
FPMAX MCSMAX	12.176
FP MAX	12.324
W	0.416
p	$7.202 \cdot 10^{-15}$

(c) *MUV*

Table F.2

Mean rankings of $EF_{FF1\%}$ of the methods, sorted in ascending order. Here, a higher rank corresponds to a higher value of $EF_{FF1\%}$, and thus a better performance

method	$\bar{x}_{F1\%}$	method	$\bar{x}_{F1\%}$
FP MAX	2.909	Tversky HS 30	3.429
FPMAX MCSMAX	3.091	Tversky HS 50	3.571
MCS MAX	3.818	Tversky HS 70	3.786
FPMAX MCSMAX HS90	4.818	Tversky HS 80	4.821
FP AVG	5.000	Tversky HS 100	5.786
FPMAX HS90	6.182	Tversky HS 95	6.107
MCSMAX HS90	7.000	Tversky HS 90	6.179
MCS AVG	7.591	FPMAX HS90	8.714
Tversky HS 95	9.636	MCSMAX HS90	9.357
Tversky HS 90	9.818	MCS AVG	9.679
Tversky HS 100	10.045	FPMAX MCSMAX HS90	10.143
Tversky HS 80	10.364	FP AVG	10.929
Tversky HS 50	12.773	FP MAX	11.786
Tversky HS 70	12.909	FPMAX MCSMAX	12.607
Tversky HS 30	14.045	MCS MAX	13.107
W	0.681	W	0.577
p	$5.604 \cdot 10^{-16}$	p	$1.396 \cdot 10^{-17}$

(a) *MDDR*(b) *WOMBAT*

method	$\bar{x}_{F1\%}$
Tversky HS 100	5.765
MCS AVG	6.176
Tversky HS 30	6.324
Tversky HS 80	6.500
Tversky HS 90	6.500
Tversky HS 95	6.794
Tversky HS 70	6.941
Tversky HS 50	7.559
FPMAX MCSMAX HS90	8.235
FP AVG	8.324
FPMAX HS90	9.059
MCSMAX HS90	9.294
FP MAX	10.500
MCS MAX	10.853
FPMAX MCSMAX	11.176
W	0.246
p	$2.049 \cdot 10^{-7}$

(c) *MUV*

Table F.3

Mean rankings of \bar{x}_{F1} of the methods, sorted in ascending order. Here, a lower rank corresponds to a higher value of \bar{x}_{F1} , and thus a worse performance

Appendix G

Activity Class performance on Weighting Schemes

Method	Activity Class											Mean Rank		Wilcoxon p	
	1	2	3	4	5	6	7	8	9	10	11	Weights	All	vs w1	vs FP MAX
HS 90 w1	0.434	0.288	0.235	0.203	0.393	0.312	0.281	0.333	0.295	0.089	0.183	4.182	4.545	1.00	$9.77 \cdot 10^{-4}$
HS 90 w2	0.062	0.061	0.08	0.087	0.453	0.201	0.105	0.172	0.262	0.046	0.256	1.818	1.818	$1.86 \cdot 10^{-2}$	$9.77 \cdot 10^{-4}$
HS 90 w3	0.297	0.215	0.228	0.194	0.452	0.274	0.236	0.252	0.225	0.082	0.204	3.182	3.364	$5.37 \cdot 10^{-2}$	$9.77 \cdot 10^{-4}$
HS 90 w4	0.204	0.172	0.183	0.187	0.563	0.311	0.209	0.25	0.318	0.069	0.251	3.091	3.364	0.28	$9.77 \cdot 10^{-4}$
HS 90 w5	0.417	0.274	0.256	0.182	0.245	0.203	0.214	0.241	0.166	0.092	0.165	2.727	3	$2.07 \cdot 10^{-2}$	$9.77 \cdot 10^{-4}$
FP HS90 w1	0.635	0.473	0.39	0.324	0.691	0.637	0.558	0.476	0.436	0.23	0.307	3.636	8.909	1.00	$8.30 \cdot 10^{-2}$
FP HS90 w2	0.198	0.192	0.236	0.238	0.73	0.546	0.333	0.316	0.458	0.11	0.363	1.818	6.182	$1.37 \cdot 10^{-2}$	$4.88 \cdot 10^{-3}$
FP HS90 w3	0.53	0.403	0.377	0.338	0.759	0.621	0.495	0.392	0.408	0.219	0.34	3	8.182	0.18	$4.88 \cdot 10^{-3}$
FP HS90 w4	0.418	0.34	0.318	0.305	0.823	0.649	0.454	0.393	0.482	0.168	0.376	3.182	8.364	0.21	$2.44 \cdot 10^{-2}$
FP HS90 w5	0.638	0.49	0.421	0.341	0.52	0.53	0.497	0.432	0.31	0.257	0.267	3.364	8.455	0.12	$1.37 \cdot 10^{-2}$
FP	0.594	0.549	0.418	0.399	0.84	0.795	0.646	0.379	0.4	0.353	0.356	1	9.818	1.00	1.00

Table G.1

BEDROC results for the MDDR dataset. Cells in bold typeface indicate that the weighting scheme that performed best for a given activity class. Cells shaded in grey indicate which method performed best for the particular column of the table. For the “Mean Rank” columns, the one titled “weights” displays the mean of the ranks comparing the weighting schemes for a given search method (thus identifying the performance of a weighting scheme for a given search method). The “all” column by contrast shows the mean of the ranks, of all the methods (thus identifying the best combination of search method and weight). The “Wilcoxon p ” columns presents the p -values for two-tailed Wilcoxon signed-rank tests, “vs w1” for the method being tested against the w1 equivalent, and “vs FP MAX” for the given method against FP MAX.

Method	Activity Class																	Mean Rank		Wilcoxon p	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Weights	All	vs w1	vs FP MAX
HS 90 w1	0.044	0.097	0.05	0.043	0	0.079	0.006	0.055	0.003	0.006	0	0	0.116	0.073	0.046	0	0	1.529	1.941	1.00	$1.53 \cdot 10^{-5}$
HS 90 w2	0.104	0.001	0.13	0.035	0.022	0.095	0.007	0.103	0.077	0.016	0.004	0.098	0.189	0.051	0.036	0.021	0.105	2.765	3.706	$3.51 \cdot 10^{-2}$	$7.63 \cdot 10^{-5}$
HS 90 w3	0.11	0.093	0.111	0.074	0.041	0.128	0.039	0.104	0.055	0.033	0.017	0.009	0.362	0.053	0.038	0.057	0.065	3.647	4.794	$3.81 \cdot 10^{-4}$	$5.48 \cdot 10^{-4}$
HS 90 w4	0.112	0.035	0.117	0.079	0.013	0.13	0.037	0.098	0.053	0.036	0.029	0.041	0.267	0.047	0.049	0.037	0.095	3.412	4.529	$2.58 \cdot 10^{-3}$	$6.52 \cdot 10^{-4}$
HS 90 w5	0.064	0.164	0.098	0.082	0.032	0.087	0.059	0.137	0.057	0.019	0.007	0.002	0.305	0.117	0.126	0.045	0.029	3.647	4.941	$1.53 \cdot 10^{-5}$	$3.05 \cdot 10^{-5}$
FP HS90 w1	0.14	0.131	0.162	0.057	0.015	0.067	0	0.088	0.134	0	0.032	0.001	0.346	0.202	0.148	0.105	0.001	1.529	4.471	1.00	$1.53 \cdot 10^{-5}$
FP HS90 w2	0.125	0.035	0.185	0.125	0.042	0.161	0.003	0.132	0.171	0.033	0.074	0.181	0.433	0.069	0.071	0.08	0.101	2.647	6.882	0.17	$1.53 \cdot 10^{-4}$
FP HS90 w3	0.163	0.19	0.205	0.161	0.066	0.154	0.008	0.183	0.119	0.025	0.095	0.09	0.543	0.154	0.162	0.135	0.073	3.588	8.294	$8.39 \cdot 10^{-4}$	$2.14 \cdot 10^{-4}$
FP HS90 w4	0.152	0.135	0.185	0.125	0.047	0.158	0.002	0.139	0.153	0.036	0.123	0.133	0.49	0.127	0.094	0.108	0.098	3.176	7.794	$1.06 \cdot 10^{-2}$	$2.90 \cdot 10^{-4}$
FP HS90 w5	0.16	0.346	0.234	0.214	0.076	0.132	0.01	0.204	0.18	0.007	0.063	0.099	0.537	0.294	0.249	0.177	0.017	4.059	8.529	$1.53 \cdot 10^{-5}$	$1.51 \cdot 10^{-3}$
FP	0.266	0.309	0.284	0.44	0.128	0.142	0.035	0.167	0.256	0.206	0.132	0.295	0.595	0.404	0.464	0.204	0.057	1	10.118	1.00	1.00

Table G.2

BEDROC results for the MUV dataset.

Method	Activity Class														Mean Rank		Wilcoxon p	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	Weights	All	vs w1	vs FP MAX
HS 90 w1	0.372	0.255	0.188	0.383	0.076	0.303	0.472	0.145	0.037	0.21	0	0.225	0.12	0.197	1.857	1.857	1.00	$1.22 \cdot 10^{-4}$
HS 90 w2	0.314	0.133	0.203	0.225	0.072	0.294	0.377	0.195	0.089	0.204	0.04	0.159	0.029	0.149	1.429	1.5	$7.85 \cdot 10^{-2}$	$1.22 \cdot 10^{-4}$
HS 90 w3	0.602	0.305	0.355	0.436	0.162	0.57	0.749	0.232	0.347	0.295	0.155	0.305	0.162	0.283	4.321	4.821	$1.22 \cdot 10^{-4}$	$1.22 \cdot 10^{-4}$
HS 90 w4	0.527	0.228	0.28	0.421	0.109	0.517	0.546	0.243	0.139	0.309	0.015	0.263	0.081	0.278	3.286	3.429	$3.05 \cdot 10^{-3}$	$1.22 \cdot 10^{-4}$
HS 90 w5	0.511	0.378	0.359	0.436	0.227	0.423	0.773	0.187	0.391	0.235	0.081	0.338	0.21	0.271	4.107	4.607	$1.22 \cdot 10^{-4}$	$1.22 \cdot 10^{-4}$
FP HS90 w1	0.727	0.38	0.497	0.532	0.25	0.595	0.727	0.298	0.094	0.472	0.001	0.365	0.34	0.306	1.643	6	1.00	$1.22 \cdot 10^{-4}$
FP HS90 w2	0.669	0.34	0.634	0.484	0.249	0.575	0.775	0.343	0.251	0.471	0.128	0.363	0.185	0.329	1.571	6.214	0.78	$1.22 \cdot 10^{-4}$
FP HS90 w3	0.875	0.465	0.654	0.707	0.449	0.805	0.873	0.395	0.538	0.578	0.32	0.512	0.434	0.455	4.321	9.536	$1.22 \cdot 10^{-4}$	$5.80 \cdot 10^{-2}$
FP HS90 w4	0.833	0.383	0.66	0.658	0.325	0.749	0.793	0.401	0.31	0.579	0.091	0.449	0.292	0.455	3.464	8.321	$3.66 \cdot 10^{-4}$	$2.44 \cdot 10^{-4}$
FP HS90 w5	0.79	0.537	0.632	0.702	0.556	0.764	0.923	0.357	0.545	0.505	0.177	0.524	0.484	0.431	4	9.143	$1.22 \cdot 10^{-4}$	$2.02 \cdot 10^{-2}$
FP	0.923	0.665	0.748	0.627	0.716	0.832	0.991	0.56	0.427	0.643	0.228	0.542	0.59	0.5	1	10.571	1.00	1.00

Table G.3

BEDROC results for the WOMBAT dataset.

Appendix H

Modular Product Heuristics

H.0.0.1 Induced Subgraphs. The Induced subgraph heuristic was originally proposed by Raymond et al. (2002a). Induced subgraph heuristics rely on applying a topological distance constraint to bonds that occur in identical subgraphs (or fragments), the fragments of which are defined by certain rules (explained later). By imposing topological distance constraints within fragments, the creation of redundant edges between modular product edges can be avoided. Consider two nodes n_1 and n_2 in the modular product, where $n_1 = (u_i, v_i)$ and $n_2 = (u_j, v_j)$, u_i and u_j are different bonds in the first molecular graph G_1 , and v_i and v_j are different bonds in the second molecular graph G_2 . When considering the creation of an edge between n_1 and n_2 in the modular product, the following rules must apply when using the induced subgraph heuristic:

- u_i and u_j must be a member of exactly one induced subgraph in G_1
- v_i and v_j must be a member of exactly one induced subgraph in G_2

If these two rules apply, then an edge is only created between nodes n_1 and n_2 if the topological distance difference is 0 (unless they are on rings of differing sizes, which is explained later). If at least one of the two above rules are violated, the induced subgraph heuristic does not apply and thus an edge can be created anyway (subject to other heuristics).

An example is depicted with C-C induced subgraphs in Figure H.1, in which the subgraphs consist only of carbon atoms bound with single bonds. Here, G_1 has two induced C-C subgraphs containing the bonds {3, 4, 5, 6, 8} and {10, 11, 12, 13, 14, 15} respectively. G_2 has two C-C induced subgraphs containing bonds {3', 4', 5', 6', 8'} and {10', 11', 15'} respectively. The following situations will help illustrate the workings of this heuristic:

- Consider the two modular product nodes (3, 3') and (6, 6').
 - bonds 3 and 6 are both present on the same subgraph in G_1
 - bonds 3' and 6' are both present on the same subgraph in G_2
 - the bond pairs (3,6) and (3',6') have a topological distance difference of 0, thus an edge is created between the modular product nodes.
- Consider another two nodes (4, 10') and (8, 11').
 - bonds 4 and 8 are both present on the same subgraph in G_1
 - bonds 10' and 11' are both present on the same subgraph in G_2
 - the bond pairs (4,8) and (10',11') also have a topological distance difference of 0, thus an edge is created between the modular product nodes.
- Consider the two nodes (8, 8') and (11,15').
 - bonds 8 and 11 are present on *different* subgraphs in G_1 . As the heuristic does not apply, an edge is formed between the two nodes, regardless of topological distance difference.
- Consider the two nodes (3, 4') and (4, 6').
 - bonds 3 and 4 are both present on the same subgraph in G_1
 - bonds 4' and 6' are both present on the same subgraph in G_2
 - the bond pairs (3,4) and (4',6') have a topological distance difference of 1, thus an edge is *not* created between the modular product nodes.

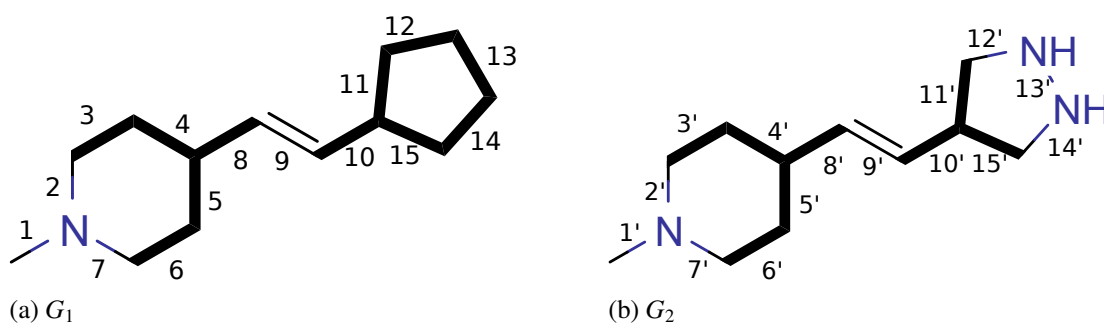


Figure H.1. Two Molecular graphs, each with two C-C induced subgraphs. Bonds comprising the subgraphs are shown in bold.

In this work, three different methods for creating induced subgraphs are defined. The first uses C-C induced subgraphs, as previously mentioned (Figure H.2b), but two additional methods for creating

induced subgraphs have also been devised. The first of these is a “rigid bond subgraph” method, which identifies and uses subgraphs which consist of ring and/or non-rotatable bonds (Figure H.2c). Bonds where both atoms are in an aromatic ring, but where the bond itself is not in a ring, are also included, as well as terminal bonds (where one atom belongs to a ring) .

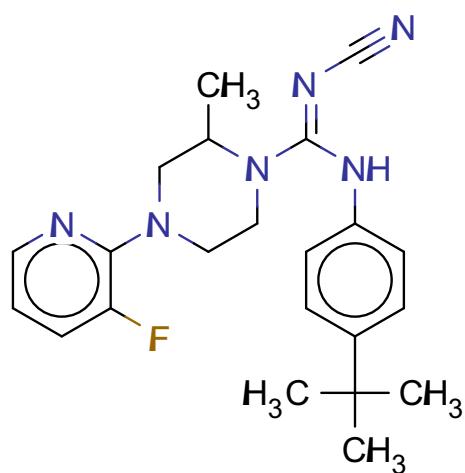
By contrast, a “non-ring bond subgraph” heuristic has been devised which acts on chains (i.e. non-ring bonds, regardless of bond order or environment), as many molecules exist with long flexible chains (Figure H.2d). The maximum path length allowed in this heuristic is the difference in the size of the longest chain (or largest shortest path length) between the two subgraphs. If this difference exceeds the size of the longest chain in either of the two subgraphs, then the maximum difference is set at the size of the shorter of the two chains. It should be noted that this does not account for branches (i.e. all bonds are included in a fragment, regardless of whether the fragment is one chain or many chains). Whilst fragments in this case could be represented as chains, instead of non-ring features with all branches included, such functionality was not implemented. We did not see the need for aligning distant branched flexible features, noting that rigid molecules have a tendency to bind proteins with greater preference to flexible molecules.

In all three of the induced subgraph heuristics, single bonds are excluded from being potential subgraphs. This is because pairs of edges are being compared for acceptance/deletion in the modular product, thus single bond subgraphs would only hinder the speed of the deletion process.

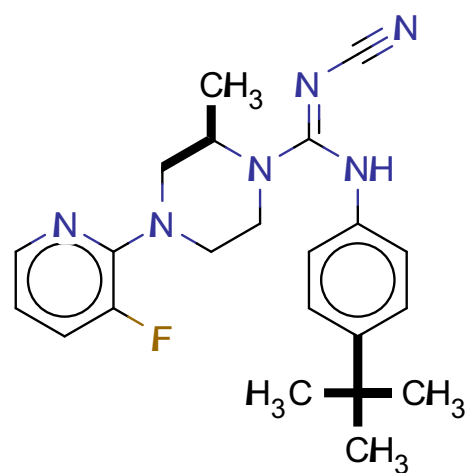
H.0.0.2 Chain Topological Distance Difference. An additional heuristic has been implemented, which simply requires that the maximum topological distance between any non-ring bond pair in a molecule cannot exceed that of the size of the longest chain in the molecule. This was implemented in an attempt to minimise redundant chain mapping (that is, bonds on one chain should only map to one other chain and not across distant chains).

H.0.0.3 Ring Heuristics. Raymond et al. (2002a) introduced the weak and strong ring heuristics as “edge deletion” heuristics. These relied on the presence of ring systems in compounds to simplify the MCS search. The weak ring heuristic would take effect between two nodes n_1 and n_2 in the modular product, where $n_1 = (u_i, v_i)$ and $n_2 = (u_j, v_j)$ (as described in the Induced Subgraphs subsection), if the following conditions were satisfied:

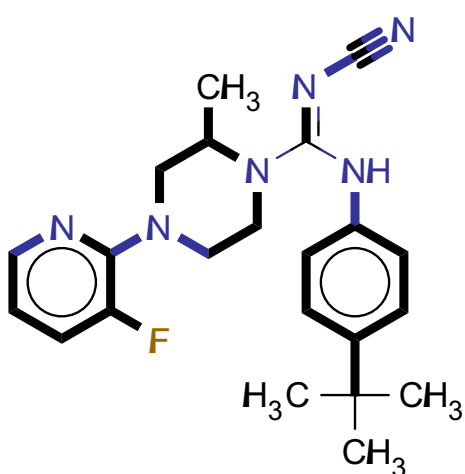
1. u_i and u_j are ring bonds in the *same ring* (in G_1). The original methodology required that the ring had seven members or less, though we have omitted this size restriction.



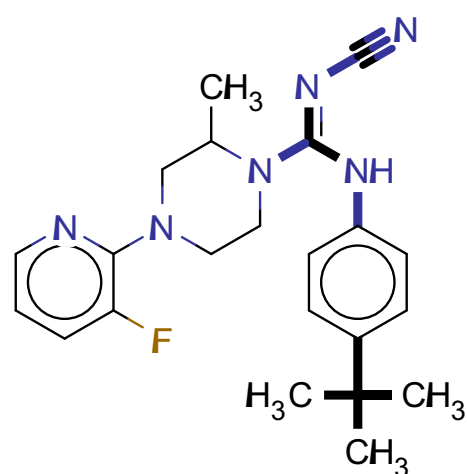
(a)



(b) C-C induced subgraphs



(c) Rigid bond induced subgraphs



(d) Non-ring induced subgraphs

Figure H.2. Induced subgraph heuristic types used in this work. Bold edges are those that comprise the relevant induced subgraphs.

2. either u_i or u_j is an unfused bond (meaning that the two bonds correspond to a single unambiguous ring).

3. v_i and v_j are ring bonds. This is regardless of whether they are in the same ring, or not (in G_2).
4. either v_i or v_j is an unfused bond.

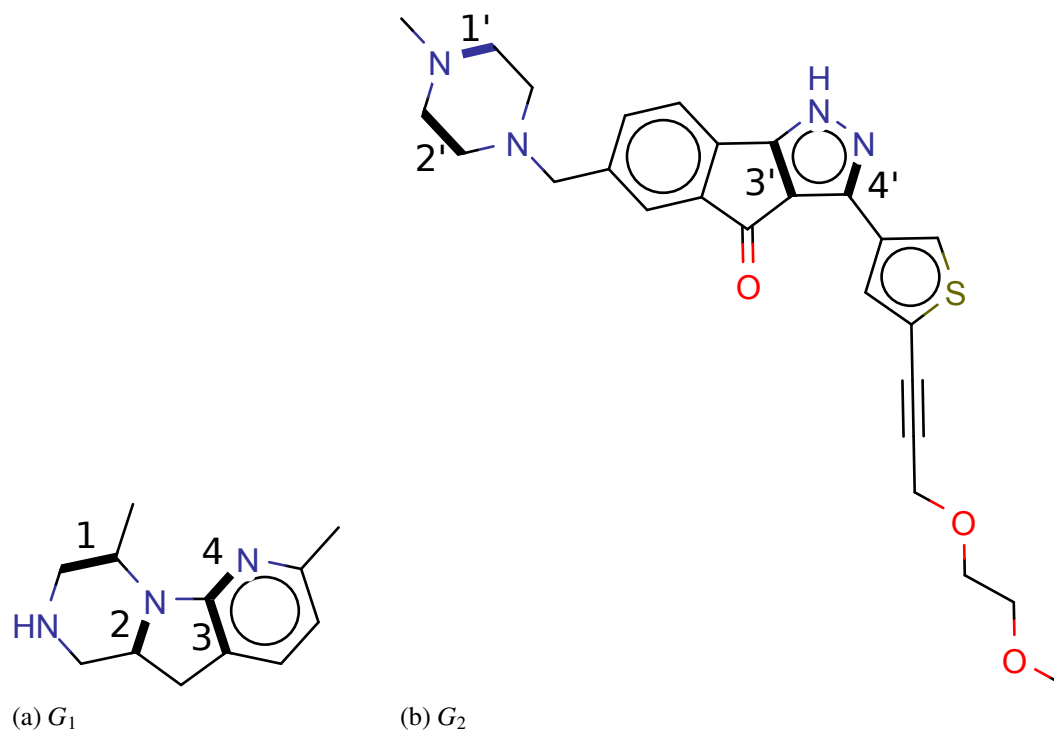


Figure H.3. An example involving the ring heuristics.

If these requirements have been met for the node pairs, the weak ring heuristic states that v_i and v_j must also be in the same ring. If this is not the case, then no edge is formed between the two node pairs in the modular product. If the weak ring heuristic is passed and both rings are aromatic, the strong ring heuristic is passed if the topological distance difference between the node pairs, is 0.

Figure H.3 can be used to demonstrate the workings of the ring heuristics.

Consider the two modular product nodes $(1, 1')$ and $(2, 2')$:

- bonds 1 and 2 are in the same ring in G_1 , and whilst 2 is a fused bond, 1 is not, thus the heuristic can still be used.
- bonds 1' and 2' are also both ring bonds, and are not fused bonds.
- as 1' and 2' are on the same ring in G_2 , the weak ring heuristic is passed.
- the strong ring heuristic does not apply as both rings need to be aromatic (of which neither is).

Next, Consider the two modular product nodes (2, 1') and (3, 2'):

- bonds 2 and 3 are in the same ring in G_1 . However, bonds 2 and 3 are both fused, so the heuristic may not apply

Finally, Consider the two modular product nodes (3, 3') and (4, 4'):

- bonds 3 and 4 are in the same ring in G_1 , and only bond 3 is a fused bond.
- bonds 3' and 4' are ring bonds, and only bond 3' is a fused bond.
- bonds 3' and 4' are in the same ring in G_2 , and thus the weak ring heuristic is passed
- however, the strong ring heuristic is failed, as the topological distance between the two node pairs are not equal. The distance between the bonds 3 and 4 is 1, whereas between 3' and 4', it is 2.

A short-coming in the strong ring heuristic arises when rings of different sizes are compared, as shown in Figure H.4. The assumption that the aromatic ring bonds must have the same path length in the strong ring heuristic, severely reduces the size of the MCES obtained in this example, as shown in Figure H.4b. To remedy this, the topological difference allowed is set to be equivalent to the difference in size (in bonds) between the two rings, rather than 0, as shown in Figure H.4c, where the distance constraint is now 1. The same situation is applied in all forms of induced subgraph heuristics where the bonds being compared are in rings of different sizes.

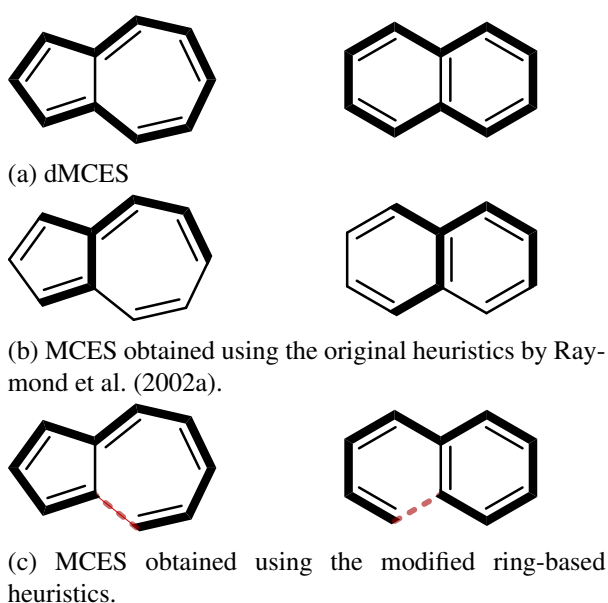


Figure H.4. Various MCEs retrieved using modular product deletion heuristics. Most bonds which are unmatched in H.4b compared to the dMCEs, are due to a topological distance greater than 0. The dotted red bond is not part of the MCEs in H.4c as this bond pair does not satisfy the weak ring heuristic (the bonds originate from different rings in each molecule).

Appendix I

MCS algorithm implementation information

I.0.0.1 VLibMCS. The unaltered code has been used from CDK for this algorithm.

I.0.0.2 SMSD. The unaltered code has been used from CDK for this algorithm.

I.0.0.3 fMCS. This has been coded from Andrew Dalke's original implementation in Python (using RDKit). We used the VF2 subgraph isomorphism algorithm as implemented in CDK to verify if a subgraph in fMCS was isomorphic. Given the speed of this subgraph isomorphism algorithm, it is uncertain if using canonical SMARTS would actually provide a speed benefit, therefore non-canonical SMARTS was stored instead of the semi-canonical SMARTS that featured in the Python implementation. SMARTS in this work was determined using AMBIT-SMARTS (Jeliazkova & Kochev, 2011), due to its high speed of processing and inherent design for use with CDK.

I.0.0.4 MaxCliqueSeq. This algorithm was coded using the original C++ source code as a guide. Whilst there are no changes to mention, it is important to note an effective lower bound that is established in the re-ordering process. As the node ordering process successively removes nodes of minimal degree first and adjusts the degrees of the other nodes accordingly with each removal, there will be a small set of nodes at the end of the processes with equal degree values. This set of vertices of equal degree forms a clique, thus presenting an initial lower bound which hastens the clique detection process. The authors in their same publication also described a parallel implementation, MaxCliquePara. This has not been implemented.

I.0.0.5 Bron-Kerbosch. The Bron-Kerbosch algorithm has been implemented in the recursive form using pivot vertices as described by Cazals and Karande (2008).

I.0.0.6 Carraghan-Paralos. This was translated from the FORTRAN code as provided by the original authors in their appendices.

I.0.0.7 RASCAL. Implementing RASCAL proved to be a challenge due to the multitude of ways in which the original methodology could be interpreted. It was implemented in this work recursively.

The Minimum Similarity Index lower bound was removed from consideration as it gave some unrealistic estimates for a lower bound (i.e. the estimate would exceed the size of one of the two graphs being compared). As we wanted the MCES between any pair of molecules regardless of their structural similarity, the lower bound in the algorithm was set to be the size of the maximum clique discovered by the algorithm at a given stage. Although we could have devised a sharper lower bound (such as one derived from path-based fingerprints), we did not do this to keep to the methodology of the original algorithm.

The initial vertex partitioning method used in the original work was not mentioned clearly. We made the assumption that each initial partition contains nodes that cannot possibly be neighbours. In this case, a partition would contain all the nodes where all the nodes in the partition would correspond to one bond in G_1 . There would therefore be a number of partitions equal to the number of bonds in G_1 . This process is outlined in Algorithm 5. Other partition-related functions have been kept as described in the original algorithm. We note however that in internal tests, the colouring algorithm has almost no effect on the upper bound (that is, the partitioning method is superior), though we still use the colouring algorithm to keep to RASCAL's original methodology. Each partition is represented as an array, so unfortunately the advantage that would have been gained using Bitstring representations is lost.

Data: Modular Product G , Molecular graph G_1

Result: Partition Set P

Function *initialisePartitions*(G, G_1):

```

|  $P = \emptyset$  ;
| for  $n$  in  $1..|E(G_1)|$  do
|   |  $P_n = \{v \in V(G) : v_1 = E(G_1)_n\}$  ;
| end

```

Algorithm 5: Initial partitioning algorithm. $v \in E(G_1)_n$ refers to the modular product node (or edge pair) v containing the identified edge in G_1 .

We have also not implemented Kikusts pruning and equivalence class pruning. In our implementation we saw no benefit in time performance when Kikusts pruning was used (if anything, it slowed

the progress down). Whilst the importance of equivalence class pruning becomes apparent when comparing molecules with large amounts of global symmetry, we simply did not have enough time to implement the heuristic.

I.0.0.8 CDKMCS. The unaltered code has been used from CDK for this algorithm.

I.0.0.9 kcombu. The authors' implementation of the algorithm sought the MCIS, whereas we are interested in the MCES, thus several adjustments were made to correct for this. The main adjustment is that bond pairs are being sought as opposed to atom pairs. The "Bond_Equivalence" and "Bond_Connection" functions have been adjusted to test for adjacent bonds, not adjacent vertices. The heuristic score $d_{neighbor}$ has been modified to count the number of different bonds around a bond's neighbourhood, as in Algorithm 6.

Data: Bond a from G_1 , Bond b from G_2

Result: score S

Function $d_{neighbor}(a, b)$:

```

for  $a_n$  in  $N(a)$  do
  |
  |  $S = 0$  ;
  | if  $\neg \exists b_n \in N(b)$  such that  $b_n \sim a_n$  then
  | |  $S = S + 1$  ;
  | end
end

```

Algorithm 6: modified $d_{neighbor}$ function. $N(a)$ represents the adjacent bonds to bond a , and \sim represents a match between two bonds.

The "Nonredundancy_By_Selection_Score" function however has been implemented largely the same as in the original work, noting that atom label frequencies are still compared (as opposed to modifying the function to account for bond label frequencies, which wasn't necessary as a match list contains both atoms and bonds).

For the d_{ec} heuristic score, the initial values assigned to bonds are the number of bonds adjacent to them, though the rest of the algorithm otherwise continues as expected. All containers containing bond pairs are implemented as arrays. The value of the constant K (the maximum number of identified common subgraphs at any one stage in the algorithm) is set to 40, as used in their original work (Kawabata, 2011).

I.0.0.10 ChemAxon_MCS. Although this is a proprietary algorithm, we note that there is an additional overhead in converting molecules from CDK format to ChemAxon format. For any time

statistics quoted, we quote the time taken *excluding this additional overhead*. To reduce the time required at this stage we use SMILES (or SMARTS for queries), though when the actual bond correspondences are required (as opposed to just the size of the MCS), we use SDF format. When using SMILES notation (as in this chapter), from internal testing, the impact on time performance is minimal.

I.0.0.11 consR. This implementation has a great number of differences from the original implementation in order to make it functional for molecular graphs. Eigendecomposition in this case was implemented using CERN’s “Colt” Java API (CERN, 2004). Most changes were implemented on suggestion of the authors, post-publication (Zhu & Yu, 2013).

The local similarity of two vertices, $S_l[u, v]$ has undergone some modifications, as shown in the equations below.

$$S_l[u, v] = \frac{(n_{min} + D(u, v))^2}{(|V(G_u^k)| + d_{sum}(G_u^k)) \times (|V(G_v^k)| + d_{sum}(G_v^k))}$$

$$D[u, v] = |L(u) \cap L(v)| + \sum_{(u', v') \in M_{uv}} w(u', v')$$

$$d_{sum}(G) = \sum_{u \in G} d(u)$$

Note that the local similarity has been adapted to account for labels, though as we wanted to allow for SMARTS matching the algorithm was modified. Notably, this manifests in the statements $|L(u) \cap L(v)|$ and any similar statement reflecting multisets. $L(u)$ would be the multiset of labels for the atoms in $N(u)$ (i.e. the vertices adjacent to u). The main change is that rather than using multisets for each matching attempt of two node neighbourhoods, a count of the matches is made for the neighbourhoods, in a similar fashion to $d_{neighbour}$ in the *kcombu* algorithm except that atoms are

matched instead of bonds. This process is detailed in Algorithm 7.

Data: Atom a from G_1 , Atom b from G_2

Result: weight w

Function *compareNeighbourLabels*(a, b):

```

|    $B = N(b)$  ; // set copy
|   for  $a'$  in  $N(a)$  do
|       |    $w = 0$  ;
|       |   if  $\exists b' \in B$  such that  $b' \sim a'$  then
|           |    $w = w + 1$  ;
|           |    $B = B \setminus b'$  ; // remove  $b'$  from  $B$ 
|       |   end
|   end
| end

```

Algorithm 7: replacement for the statement $|L(u) \cap L(v)|$. $N(a)$ represents the adjacent bonds to bond a , and \sim represents a match between two bonds.

The anchor selection process has been modified. Firstly, we did not specify an arbitrary similarity threshold to select pairs as anchors (what if the threshold is high to the point where no pairs are suitable?). Thus, the pairs are instead sorted in descending order and the top few are chosen (specified by an arbitrary constant). Secondly, the larger average degree constraint δ , is highly unsuitable when comparing graphs with highly different edge densities. Thus, if the value of δ exceeds the maximum degree of a node in one of the two graphs, it is set to said maximum degree.

After one round of anchor expansion, there may be some nodes in both graphs which are unmatched. A modification implemented involves selecting the unmatched pair with the highest local similarity, and adding it to the candidate list, so that the expansion process can continue. This process continues until there are no unmatched nodes in one of the two graphs.

The random minimal vertex cover algorithm was replaced with a more recent algorithm for finding a minimum vertex cover (Ugurlu, 2012). The goal of using this was simply to remove the sole cause of random error in the original work, and thus minimising the number of iterations required for finding vertex covers. This vertex cover algorithm has also shown to yield accurate approximations of the minimum vertex cover in very small time frames for even large graphs (less than one second).

On running the algorithm for MCS finding, the algorithm is executed for 1, 2 and then for 3 anchors, with a value of 2 for the neighbourhood size constant k when determining the local similarity.

Appendix J

Modular Product Information

pair	dMCS			hMCS			tdMCS ($\theta = 2$)			tdMCS ($\theta = 0$)		
	Nodes	Density	Time	Nodes	Density	Time	Nodes	Density	Time	Nodes	Density	Time
M1	416	0.675	0.010 ± 0.001	416	0.319	0.026 ± 0.002	416	0.496	0.011 ± 0.002	416	0.144	0.009 ± 0.003
M2	512	0.711	0.014 ± 0.001	512	0.303	0.034 ± 0.003	512	0.549	0.012 ± 0.001	512	0.163	0.013 ± 0.002
M3	2,257	0.805	0.234 ± 0.050	2,257	0.085	0.928 ± 0.122	2,257	0.374	0.174 ± 0.021	2,257	0.087	0.154 ± 0.019
N1	630	0.793	0.021 ± 0.001	630	0.572	0.034 ± 0.009	630	0.274	0.021 ± 0.001	630	0.064	0.019 ± 0.004
N2	264	0.738	0.004 ± 0.000	264	0.352	0.006 ± 0.003	264	0.223	0.004 ± 0.000	264	0.055	0.004 ± 0.001
N3	336	0.734	0.007 ± 0.001	336	0.603	0.010 ± 0.001	336	0.247	0.007 ± 0.002	336	0.059	0.006 ± 0.001
S1	416	0.805	0.012 ± 0.003	408	0.697	0.017 ± 0.001	416	0.309	0.010 ± 0.001	416	0.087	0.008 ± 0.003
S2	448	0.837	0.011 ± 0.001	440	0.764	0.021 ± 0.004	448	0.311	0.012 ± 0.001	448	0.083	0.011 ± 0.003
S3	341	0.778	0.007 ± 0.001	341	0.095	0.010 ± 0.001	341	0.361	0.008 ± 0.001	341	0.095	0.007 ± 0.002
S4	457	0.828	0.012 ± 0.002	361	0.760	0.014 ± 0.001	457	0.349	0.013 ± 0.002	457	0.085	0.011 ± 0.001
S5	1,015	0.867	0.053 ± 0.006	617	0.810	0.033 ± 0.004	1,015	0.267	0.054 ± 0.004	1,015	0.061	0.039 ± 0.007
S6	1,520	0.869	0.121 ± 0.014	812	0.271	0.110 ± 0.009	1,520	0.378	0.118 ± 0.011	1,520	0.085	0.066 ± 0.004

Table J.1

Modular product information for the *S*, *N* and *M* compound pairs. Construction time, as well as the number of nodes and the density of the modular product, are shown per pair, for each MCS type (dMCS, hMCS and the two tdMCS types). Time (mean ± standard deviation) is displayed in seconds, although they have been recorded to millisecond precision. The bold value per row indicates the best size, or fastest time, achieved out of the algorithms.

pair	dMCS			hMCS			tdMCS ($\theta = 2$)			tdMCS ($\theta = 0$)		
	Nodes	Density	Time	Nodes	Density	Time	Nodes	Density	Time	Nodes	Density	Time
80a	99	0.706	0.001 ± 0.000	69	0.518	0.001 ± 0.000	99	0.612	0.001 ± 0.000	99	0.232	0.001 ± 0.000
89a	287	0.790	0.004 ± 0.001	287	0.652	0.011 ± 0.003	287	0.429	0.004 ± 0.000	287	0.122	0.003 ± 0.001
2a	278	0.794	0.004 ± 0.000	278	0.717	0.008 ± 0.003	278	0.385	0.004 ± 0.001	278	0.119	0.004 ± 0.001
57a	267	0.668	0.003 ± 0.001	267	0.250	0.008 ± 0.001	267	0.568	0.004 ± 0.001	267	0.192	0.003 ± 0.001
92a	525	0.703	0.013 ± 0.003	525	0.242	0.024 ± 0.008	525	0.506	0.011 ± 0.002	525	0.149	0.009 ± 0.001
31a	263	0.663	0.004 ± 0.001	263	0.260	0.009 ± 0.001	263	0.572	0.004 ± 0.001	263	0.195	0.003 ± 0.001
19a	103	0.670	0.001 ± 0.000	48	0.574	0.001 ± 0.000	103	0.481	0.001 ± 0.000	103	0.175	0.001 ± 0.000
100a	654	0.746	0.018 ± 0.003	654	0.387	0.028 ± 0.005	654	0.479	0.020 ± 0.005	654	0.131	0.015 ± 0.004
91a	176	0.676	0.001 ± 0.000	90	0.279	0.002 ± 0.000	176	0.503	0.002 ± 0.000	176	0.149	0.001 ± 0.000
94a	115	0.691	0.001 ± 0.000	103	0.623	0.001 ± 0.000	115	0.412	0.001 ± 0.000	115	0.138	0.001 ± 0.000
72a	142	0.747	0.002 ± 0.001	126	0.642	0.002 ± 0.000	142	0.612	0.001 ± 0.000	142	0.201	0.002 ± 0.000
18a	113	0.696	0.001 ± 0.000	58	0.509	0.001 ± 0.000	113	0.464	0.001 ± 0.000	113	0.161	0.001 ± 0.000
34a	80	0.611	0.000 ± 0.000	50	0.387	0.001 ± 0.000	80	0.295	0.000 ± 0.000	80	0.131	0.000 ± 0.000
47a	179	0.751	0.002 ± 0.000	161	0.432	0.004 ± 0.001	179	0.446	0.002 ± 0.000	179	0.139	0.002 ± 0.000
69a	172	0.715	0.002 ± 0.000	142	0.568	0.002 ± 0.001	172	0.424	0.002 ± 0.000	172	0.130	0.001 ± 0.000
28a	97	0.620	0.001 ± 0.000	37	0.584	0.000 ± 0.000	97	0.292	0.001 ± 0.000	97	0.108	0.001 ± 0.000
96a	646	0.736	0.020 ± 0.006	646	0.348	0.035 ± 0.010	646	0.482	0.017 ± 0.003	646	0.131	0.015 ± 0.005
50a	96	0.594	0.001 ± 0.000	42	0.616	0.001 ± 0.000	96	0.417	0.001 ± 0.000	96	0.153	0.000 ± 0.000
8a	147	0.649	0.001 ± 0.000	47	0.710	0.001 ± 0.000	147	0.438	0.001 ± 0.000	147	0.123	0.001 ± 0.000
13a	91	0.540	0.000 ± 0.000	31	0.647	0.001 ± 0.000	91	0.271	0.001 ± 0.000	91	0.106	0.001 ± 0.001
45a	68	0.603	0.000 ± 0.000	68	0.578	0.001 ± 0.000	68	0.447	0.001 ± 0.000	68	0.167	0.000 ± 0.000
12a	152	0.692	0.001 ± 0.000	122	0.539	0.002 ± 0.000	152	0.311	0.001 ± 0.000	152	0.110	0.001 ± 0.000
30a	92	0.676	0.001 ± 0.000	62	0.542	0.001 ± 0.000	92	0.218	0.001 ± 0.000	92	0.084	0.001 ± 0.000
20a	345	0.659	0.005 ± 0.000	345	0.207	0.012 ± 0.003	345	0.508	0.007 ± 0.000	345	0.158	0.005 ± 0.001
88a	27	0.524	0.000 ± 0.000	27	0.484	0.000 ± 0.000	27	0.459	0.000 ± 0.000	27	0.160	0.000 ± 0.000

Table J.2

Modular product information for the Franco compound pairs.

Appendix K

S, N and M Compound Pair Performances

pair	BK			CP	MaxCliqueSeq			RASCAL			kcombu			consR		CA_MCS	
	Size	Time	Size		Time	Size	Time	Size	Time	Size	Time	Size	Time	Size	Time		
M1	12	>300	16	>300	16	0.142 ± 0.016	16	133.986 ± 4.816	12	0.038 ± 0.016	11	0.043 ± 0.017	16	0.053 ± 0.004			
M2	18	>300	19	>300	20	22.892 ± 1.174	20	111.823 ± 8.272	15	0.074 ± 0.007	11	0.066 ± 0.023	20	0.065 ± 0.004			
M3	18	>300	18	>300	24	>300	25	>300	14	0.227 ± 0.029	16	0.083 ± 0.016	24	0.210 ± 0.025			
N1	19	>300	23	>300	25	>300	24	>300	22	0.088 ± 0.010	19	0.048 ± 0.010	24	0.099 ± 0.012			
N2	12	>300	16	191.712 ± 8.708	16	0.346 ± 0.027	16	2.282 ± 0.184	14	0.028 ± 0.004	11	0.019 ± 0.004	16	0.009 ± 0.004			
N3	11	>300	12	>300	12	0.961 ± 0.061	11	>300	11	0.037 ± 0.013	7	0.041 ± 0.010	12	0.041 ± 0.006			
S1	24	>300	32	>300	32	0.224 ± 0.038	32	0.306 ± 0.070	32	0.014 ± 0.007	21	0.045 ± 0.027	32	0.026 ± 0.007			
S2	24	>300	33	>300	35	0.447 ± 0.062	35	0.277 ± 0.085	34	0.025 ± 0.004	31	0.080 ± 0.017	35	0.014 ± 0.004			
S3	23	>300	33	>300	33	0.049 ± 0.004	33	0.102 ± 0.019	33	0.013 ± 0.004	33	0.065 ± 0.011	33	0.007 ± 0.002			
S4	26	>300	25	>300	33	138.095 ± 3.898	31	>300	27	0.111 ± 0.025	24	0.071 ± 0.012	33	0.044 ± 0.011			
S5	26	>300	33	>300	44	>300	42	>300	45	0.119 ± 0.021	29	0.137 ± 0.019	45	0.098 ± 0.010			
S6	28	>300	43	>300	54	>300	54	>300	54	0.186 ± 0.040	40	0.180 ± 0.027	54	0.061 ± 0.021			

Table K.1

dMCS performance information for the S, N and M compound pairs. Modular product information and algorithm performance for each compound pair. The columns under the “Modular Product” heading show the time taken to construct the modular product between a pair of molecular graphs, as well as the number of nodes and the density of the modular product. Subsequent headings denote the MCS algorithm, where search time and MCS size are shown. Time (mean ± standard deviation) is displayed in seconds, although they have been recorded to millisecond precision. The bold value per row indicates the best size, or fastest time, achieved out of the algorithms. If a time quoted is in italic font, then the mean was calculated excluding any results which failed to run to completion. Results which only possess non-completions are also in italic font (the standard deviation being fixed to 0 as it was not calculated).

pair	Size	BK		CP	MaxCliqueSeq			RASCAL			kcombu	
		Time	Size		Time	Size	Time	Size	Time	Size	Time	
M1	12	>300	12	0.966 ± 0.053	12	0.056 ± 0.011	12	4.814 ± 0.269	12	0.044 ± 0.006		
M2	12	>300	16	2.530 ± 0.100	16	0.082 ± 0.018	16	6.412 ± 0.614	15	0.081 ± 0.008		
M3	25	>300	25	1.723 ± 0.114	25	1.250 ± 0.199	24	5.115 ± 0.421	14	0.246 ± 0.032		
N1	16	>300	22	>300	23	77.904 ± 0.633	23	22.453 ± 2.302	22	0.079 ± 0.008		
N2	14	87.373 ± 1.200	14	0.402 ± 0.023	14	0.025 ± 0.011	14	0.142 ± 0.024	14	0.027 ± 0.011		
N3	11	>300	12	62.151 ± 2.076	12	2.092 ± 0.400	12	88.531 ± 2.643	11	0.038 ± 0.003		
S1	17	>300	32	>300	32	0.157 ± 0.021	32	0.200 ± 0.039	32	0.010 ± 0.003		
S2	21	>300	33	>300	35	3.832 ± 0.669	35	0.160 ± 0.033	34	0.042 ± 0.014		
S3	33	0.184 ± 0.023	33	0.095 ± 0.013	33	0.017 ± 0.006	33	0.022 ± 0.005	33	0.013 ± 0.003		
S4	27	>300	28	>300	33	1.111 ± 0.165	33	135.161 ± 6.429	29	0.077 ± 0.021		
S5	32	>300	34	>300	45	18.756 ± 0.275	43	>300	45	0.095 ± 0.027		
S6	17	>300	54	>300	54	0.189 ± 0.024	54	0.894 ± 0.053	54	0.086 ± 0.013		

Table K.2

hMCS modular product and MCS performance information for the S, N and M compound pairs.

pair	BK		CP		MaxCliqueSeq		RASCAL		kcombu	
	Size	Time	Size	Time	Size	Time	Size	Time	Size	Time
M1	12	>300	14	10.785 ± 0.391	14	0.233 ± 0.019	14	10.541 ± 1.024	12	0.044 ± 0.010
M2	17	>300	18	>300	18	2.655 ± 0.253	18	29.346 ± 1.610	16	0.070 ± 0.010
M3	17	>300	24	>300	24	>300	23	>300	22	0.425 ± 0.036
N1	18	>300	21	240.155 ± 11.710	21	0.317 ± 0.082	21	1.096 ± 0.071	19	0.116 ± 0.030
N2	14	11.088 ± 0.472	14	0.055 ± 0.018	14	0.012 ± 0.002	14	0.035 ± 0.012	13	0.025 ± 0.003
N3	11	53.913 ± 1.517	11	0.068 ± 0.004	11	0.021 ± 0.006	11	0.171 ± 0.035	9	0.037 ± 0.009
S1	12	>300	32	>300	32	0.042 ± 0.014	32	0.083 ± 0.024	32	0.019 ± 0.001
S2	11	>300	32	>300	34	0.049 ± 0.006	34	0.180 ± 0.041	34	0.040 ± 0.011
S3	16	>300	33	59.266 ± 3.943	33	0.027 ± 0.002	33	0.035 ± 0.011	33	0.013 ± 0.005
S4	15	>300	29	104.672 ± 2.360	29	0.041 ± 0.017	29	0.424 ± 0.060	27	0.118 ± 0.010
S5	15	>300	25	>300	37	0.264 ± 0.036	37	2.369 ± 0.225	36	0.140 ± 0.013
S6	26	>300	48	>300	54	215.023 ± 6.062	54	8.441 ± 2.093	54	0.145 ± 0.022

Table K.3

tdMCS ($\theta = 2$) modular product and MCS performance information for the *S*, *N* and *M* compound pairs.

pair	BK		CP		MaxCliqueSeq		RASCAL		kcombu	
	Size	Time	Size	Time	Size	Time	Size	Time	Size	Time
M1	10	3.413 ± 0.133	10	0.018 ± 0.006	10	0.017 ± 0.003	10	0.071 ± 0.018	9	0.043 ± 0.012
M2	14	13.540 ± 1.116	14	0.037 ± 0.010	14	0.032 ± 0.004	14	0.137 ± 0.019	11	0.052 ± 0.007
M3	25	>300	25	1.202 ± 0.211	25	0.400 ± 0.024	24	2.453 ± 0.047	20	0.517 ± 0.039
N1	13	2.801 ± 0.061	13	0.052 ± 0.013	13	0.032 ± 0.005	13	0.091 ± 0.016	13	0.074 ± 0.019
N2	13	0.040 ± 0.012	13	0.005 ± 0.001	13	0.007 ± 0.001	13	0.006 ± 0.001	13	0.029 ± 0.015
N3	7	0.099 ± 0.007	7	0.009 ± 0.001	7	0.010 ± 0.003	7	0.013 ± 0.002	6	0.030 ± 0.005
S1	22	3.444 ± 0.251	22	0.729 ± 0.149	22	0.019 ± 0.009	22	0.041 ± 0.010	22	0.032 ± 0.012
S2	22	4.561 ± 0.397	22	0.613 ± 0.110	22	0.021 ± 0.010	22	0.048 ± 0.007	22	0.045 ± 0.014
S3	33	0.181 ± 0.013	33	0.078 ± 0.023	33	0.013 ± 0.005	33	0.018 ± 0.007	33	0.017 ± 0.003
S4	25	0.503 ± 0.007	25	0.057 ± 0.026	25	0.019 ± 0.007	25	0.036 ± 0.002	24	0.091 ± 0.035
S5	17	15.803 ± 2.855	35	12.749 ± 0.317	35	0.070 ± 0.017	35	0.128 ± 0.028	35	0.110 ± 0.012
S6	10	>300	54	229.991 ± 5.440	54	0.184 ± 0.039	54	0.243 ± 0.037	54	0.176 ± 0.016

Table K.4

tdMCS ($\theta = 0$) modular product and MCS performance information for the *S*, *N* and *M* compound pairs.

pair	VLibMCS		SMSD		fMCS		CDKMCS		BK		kcombu		CA_MCS	
	Size	Time	Size	Time	Size	Time	Size	Time	Size	Time	Size	Time	Size	Time
M1	13	1.984 ± 0.730	13	2.809 ± 1.071	13	0.018 ± 0.005	12	0.398 ± 0.265	13	0.650 ± 0.296	13	0.069 ± 0.014	13	0.023 ± 0.001
M2	19	11.280 ± 3.182	19	11.464 ± 4.162	19	2.595 ± 0.054	20	0.388 ± 0.317	20	7.739 ± 0.552	12	0.114 ± 0.019	20	0.044 ± 0.007
M3	0	>300	0	>300	25	0.011 ± 0.008	24	1.059 ± 0.151	25	>300	23	0.865 ± 0.122	24	0.136 ± 0.009
N1	7	6.179 ± 1.350	7	5.876 ± 1.416	9	0.047 ± 0.003	9	0.160 ± 0.012	9	0.082 ± 0.047	8	0.080 ± 0.014	8	0.028 ± 0.007
N2	5	1.316 ± 0.657	5	1.309 ± 0.463	5	0.001 ± 0.000	5	0.013 ± 0.004	5	0.009 ± 0.003	5	0.013 ± 0.002	5	0.008 ± 0.001
N3	4	1.725 ± 0.554	4	1.909 ± 1.004	4	0.002 ± 0.000	4	0.016 ± 0.005	4	0.024 ± 0.007	3	0.025 ± 0.008	2	0.008 ± 0.001
S1	17	1.456 ± 0.469	17	1.821 ± 0.524	17	0.102 ± 0.019	17	0.128 ± 0.063	17	0.092 ± 0.016	17	0.018 ± 0.010	17	0.017 ± 0.003
S2	14	2.918 ± 0.612	14	3.401 ± 1.222	14	0.003 ± 0.001	14	0.155 ± 0.073	14	0.065 ± 0.027	14	0.017 ± 0.003	14	0.019 ± 0.004
S3	33	2.463 ± 0.931	33	2.292 ± 0.846	33	0.004 ± 0.001	33	0.020 ± 0.001	33	3.846 ± 0.321	33	0.016 ± 0.000	33	0.004 ± 0.001
S4	25	6.157 ± 1.976	25	4.498 ± 0.943	25	15.778 ± 2.931	25	0.165 ± 0.072	25	0.227 ± 0.059	24	0.065 ± 0.011	25	0.023 ± 0.001
S5	37	12.464 ± 5.604	37	14.340 ± 6.429	37	2.267 ± 0.202	37	0.339 ± 0.087	37	18.554 ± 3.036	37	0.101 ± 0.025	37	0.049 ± 0.015
S6	52	74.290 ± 52.748	52	84.144 ± 35.562	52	0.534 ± 0.068	40	0.714 ± 0.341	52	>300	52	0.148 ± 0.012	52	0.072 ± 0.015

Table K.5

cMCS performance information for the S, N and M compound pairs. CA_MCS refers to ChemAxon_MCS, and kcombu was used to calculate the cMCS.

Appendix L

Franco Compound Pair Performances

pair	BK		CP		MaxCliqueSeq		RASCAL		kcombu		consR		CA_MCS	
	Size	Time	Size	Time	Size	Time	Size	Time	Size	Time	Size	Time	Size	Time
80a	19	21.270 ± 0.603	19	0.029 ± 0.001	19	0.004 ± 0.001	19	0.004 ± 0.001	19	0.004 ± 0.001	19	0.007 ± 0.001	19	0.004 ± 0.001
89a	23	>300	32	>300	32	0.026 ± 0.005	32	0.062 ± 0.016	32	0.009 ± 0.004	28	0.022 ± 0.003	32	0.006 ± 0.001
2a	24	>300	31	>300	34	0.024 ± 0.006	34	0.043 ± 0.015	33	0.028 ± 0.011	34	0.029 ± 0.008	34	0.006 ± 0.003
57a	20	>300	23	32.364 ± 0.963	23	0.024 ± 0.004	23	0.033 ± 0.010	20	0.010 ± 0.002	23	0.013 ± 0.003	23	0.005 ± 0.000
92a	15	>300	21	>300	28	0.143 ± 0.030	27	0.878 ± 0.222	28	0.025 ± 0.009	28	0.025 ± 0.010	28	0.008 ± 0.002
31a	14	>300	23	28.081 ± 1.382	23	0.024 ± 0.005	23	0.036 ± 0.009	23	0.010 ± 0.003	23	0.021 ± 0.010	23	0.005 ± 0.001
19a	15	9.475 ± 0.185	15	0.021 ± 0.001	15	0.004 ± 0.001	15	0.022 ± 0.009	14	0.006 ± 0.001	14	0.011 ± 0.005	15	0.004 ± 0.001
100a	16	>300	24	>300	33	0.324 ± 0.038	33	14.422 ± 0.471	29	0.055 ± 0.010	33	0.035 ± 0.009	33	0.051 ± 0.007
91a	19	>300	21	1.183 ± 0.095	21	0.009 ± 0.003	21	0.015 ± 0.005	21	0.015 ± 0.004	21	0.011 ± 0.004	21	0.004 ± 0.001
94a	13	31.797 ± 0.658	17	0.066 ± 0.011	17	0.005 ± 0.002	17	0.007 ± 0.003	15	0.005 ± 0.001	14	0.009 ± 0.002	17	0.003 ± 0.001
72a	18	>300	25	0.645 ± 0.050	25	0.006 ± 0.001	25	0.024 ± 0.004	21	0.013 ± 0.004	25	0.021 ± 0.006	25	0.008 ± 0.003
18a	13	41.554 ± 0.323	15	0.083 ± 0.007	15	0.004 ± 0.001	15	0.011 ± 0.001	13	0.009 ± 0.001	12	0.012 ± 0.003	15	0.003 ± 0.001
34a	12	0.249 ± 0.005	12	0.003 ± 0.000	12	0.002 ± 0.001	12	0.003 ± 0.001	12	0.004 ± 0.001	6	0.007 ± 0.001	12	0.002 ± 0.001
47a	16	>300	20	8.785 ± 0.671	20	0.021 ± 0.001	20	0.253 ± 0.063	19	0.026 ± 0.005	20	0.017 ± 0.004	20	0.018 ± 0.001
69a	19	>300	19	5.495 ± 0.466	19	0.033 ± 0.008	19	0.611 ± 0.056	16	0.019 ± 0.005	13	0.011 ± 0.001	19	0.017 ± 0.002
28a	10	1.616 ± 0.081	10	0.013 ± 0.000	10	0.003 ± 0.001	10	0.005 ± 0.002	10	0.009 ± 0.002	10	0.008 ± 0.003	10	0.007 ± 0.001
96a	17	>300	24	>300	29	4.017 ± 0.550	29	1.063 ± 0.153	18	0.093 ± 0.019	27	0.026 ± 0.003	29	0.045 ± 0.003
50a	11	1.262 ± 0.053	11	0.008 ± 0.000	11	0.003 ± 0.001	11	0.004 ± 0.002	8	0.003 ± 0.001	6	0.009 ± 0.002	11	0.003 ± 0.001
8a	11	62.522 ± 1.518	13	0.234 ± 0.003	13	0.007 ± 0.002	13	0.301 ± 0.021	13	0.013 ± 0.004	11	0.011 ± 0.003	13	0.013 ± 0.003
13a	8	0.142 ± 0.009	8	0.004 ± 0.001	8	0.003 ± 0.001	8	0.051 ± 0.013	6	0.005 ± 0.001	5	0.007 ± 0.001	8	0.006 ± 0.001
45a	9	0.390 ± 0.036	9	0.003 ± 0.000	9	0.002 ± 0.000	9	0.002 ± 0.000	8	0.005 ± 0.001	4	0.008 ± 0.001	8	0.006 ± 0.002
12a	12	>300	14	1.587 ± 0.021	14	0.014 ± 0.004	14	1.848 ± 0.290	13	0.018 ± 0.006	7	0.010 ± 0.003	14	0.013 ± 0.001
30a	10	4.870 ± 0.302	10	0.024 ± 0.001	10	0.004 ± 0.001	10	0.080 ± 0.021	8	0.009 ± 0.004	6	0.011 ± 0.001	10	0.008 ± 0.001
20a	14	>300	16	98.177 ± 6.472	16	0.279 ± 0.022	16	16.244 ± 1.107	12	0.024 ± 0.002	10	0.020 ± 0.006	16	0.031 ± 0.004
88a	6	0.002 ± 0.001	6	0.001 ± 0.001	6	0.001 ± 0.000	6	0.001 ± 0.000	6	0.001 ± 0.000	4	0.003 ± 0.001	5	0.002 ± 0.000

Table L.1

dMCS modular product and MCS performance information for the Franco compound pairs.

pair	BK		CP		MaxCliqueSeq		RASCAL		kcombu	
	Size	Time	Size	Time	Size	Time	Size	Time	Size	Time
80a	19	0.053 ± 0.006	19	0.002 ± 0.001	19	0.002 ± 0.001	19	0.004 ± 0.001	19	0.003 ± 0.001
89a	21	>300	32	21.107 ± 4.134	32	0.027 ± 0.007	32	0.042 ± 0.004	32	0.013 ± 0.001
2a	20	>300	34	51.832 ± 4.314	34	0.026 ± 0.004	34	0.031 ± 0.006	33	0.024 ± 0.001
57a	23	4.038 ± 0.346	23	0.022 ± 0.002	23	0.012 ± 0.003	22	0.013 ± 0.003	20	0.011 ± 0.004
92a	17	>300	28	0.182 ± 0.033	28	0.049 ± 0.009	28	0.055 ± 0.009	28	0.016 ± 0.001
31a	23	3.078 ± 0.351	23	0.023 ± 0.007	23	0.015 ± 0.006	23	0.015 ± 0.005	23	0.008 ± 0.001
19a	15	0.011 ± 0.004	15	0.001 ± 0.000	15	0.002 ± 0.001	15	0.002 ± 0.001	15	0.003 ± 0.001
100a	15	>300	33	>300	33	0.101 ± 0.016	33	0.567 ± 0.039	29	0.065 ± 0.015
91a	21	0.032 ± 0.004	21	0.002 ± 0.001	21	0.004 ± 0.001	20	0.004 ± 0.001	21	0.009 ± 0.003
94a	17	2.731 ± 0.134	17	0.014 ± 0.001	17	0.004 ± 0.002	17	0.005 ± 0.001	16	0.009 ± 0.002
72a	25	18.715 ± 1.418	25	0.030 ± 0.003	25	0.006 ± 0.003	25	0.014 ± 0.006	23	0.013 ± 0.001
18a	15	0.015 ± 0.001	15	0.002 ± 0.000	15	0.002 ± 0.001	15	0.003 ± 0.000	15	0.003 ± 0.000
34a	12	0.002 ± 0.000	12	0.001 ± 0.000	12	0.002 ± 0.001	12	0.002 ± 0.000	12	0.003 ± 0.001
47a	20	12.389 ± 0.548	20	0.029 ± 0.005	20	0.009 ± 0.001	20	0.020 ± 0.007	20	0.024 ± 0.002
69a	15	6.213 ± 0.265	19	0.023 ± 0.003	19	0.006 ± 0.001	19	0.169 ± 0.031	16	0.022 ± 0.008
28a	9	0.005 ± 0.001	9	0.001 ± 0.000	9	0.001 ± 0.000	9	0.003 ± 0.002	10	0.005 ± 0.001
96a	27	>300	29	35.001 ± 4.822	29	0.092 ± 0.012	29	0.340 ± 0.030	18	0.093 ± 0.020
50a	10	0.028 ± 0.013	10	0.001 ± 0.000	10	0.001 ± 0.000	10	0.002 ± 0.001	8	0.002 ± 0.001
8a	13	0.070 ± 0.007	13	0.002 ± 0.000	13	0.002 ± 0.000	13	0.016 ± 0.003	11	0.007 ± 0.001
13a	8	0.003 ± 0.000	8	0.001 ± 0.000	8	0.001 ± 0.000	8	0.002 ± 0.001	6	0.002 ± 0.001
45a	9	0.303 ± 0.043	9	0.003 ± 0.001	9	0.002 ± 0.001	9	0.002 ± 0.001	8	0.006 ± 0.002
12a	14	1.529 ± 0.072	14	0.015 ± 0.002	14	0.004 ± 0.001	14	0.165 ± 0.017	12	0.012 ± 0.003
30a	8	0.046 ± 0.002	8	0.002 ± 0.000	8	0.002 ± 0.001	8	0.007 ± 0.002	7	0.006 ± 0.001
20a	13	4.536 ± 0.214	13	0.032 ± 0.004	13	0.018 ± 0.003	13	0.124 ± 0.028	12	0.031 ± 0.012
88a	6	0.002 ± 0.000	6	0.001 ± 0.000	6	0.001 ± 0.000	6	0.001 ± 0.000	6	0.002 ± 0.001

Table L.2

hmCS modular product and MCS performance information for the Franco compound pairs.

pair	BK		CP		MaxCliqueSeq		RASCAL		kcombu	
	Size	Time	Size	Time	Size	Time	Size	Time	Size	Time
100a	16	>300	20	>300	33	0.101 ± 0.028	33	0.190 ± 0.014	29	0.096 ± 0.007
12a	11	2.197 ± 0.100	11	0.014 ± 0.001	11	0.005 ± 0.001	11	0.024 ± 0.007	11	0.028 ± 0.010
13a	7	0.029 ± 0.004	7	0.002 ± 0.001	7	0.002 ± 0.000	7	0.005 ± 0.003	7	0.006 ± 0.001
18a	15	0.955 ± 0.099	15	0.006 ± 0.003	15	0.003 ± 0.001	15	0.005 ± 0.001	15	0.015 ± 0.001
19a	15	0.688 ± 0.080	15	0.004 ± 0.001	15	0.004 ± 0.000	15	0.015 ± 0.005	15	0.012 ± 0.004
20a	12	>300	15	5.349 ± 0.076	15	0.211 ± 0.033	15	2.994 ± 0.760	11	0.043 ± 0.013
28a	8	0.088 ± 0.011	8	0.002 ± 0.000	8	0.003 ± 0.001	8	0.002 ± 0.000	8	0.007 ± 0.001
2a	31	>300	34	>300	34	0.018 ± 0.003	34	0.017 ± 0.002	34	0.024 ± 0.005
30a	8	0.027 ± 0.001	8	0.002 ± 0.000	8	0.003 ± 0.001	8	0.003 ± 0.001	8	0.018 ± 0.008
31a	20	>300	23	1.298 ± 0.103	23	0.017 ± 0.003	23	0.026 ± 0.002	23	0.011 ± 0.004
34a	12	0.064 ± 0.007	12	0.001 ± 0.000	12	0.002 ± 0.001	12	0.002 ± 0.000	12	0.004 ± 0.001
45a	9	0.070 ± 0.016	9	0.001 ± 0.000	9	0.002 ± 0.001	9	0.002 ± 0.001	6	0.005 ± 0.001
47a	20	32.289 ± 1.378	20	0.075 ± 0.004	20	0.006 ± 0.002	20	0.026 ± 0.006	20	0.032 ± 0.006
50a	10	0.336 ± 0.054	10	0.002 ± 0.000	10	0.003 ± 0.001	10	0.004 ± 0.001	9	0.010 ± 0.001
57a	21	>300	23	1.570 ± 0.088	23	0.021 ± 0.006	23	0.022 ± 0.007	23	0.020 ± 0.007
69a	19	44.139 ± 1.698	19	0.039 ± 0.007	19	0.006 ± 0.002	19	0.085 ± 0.022	15	0.021 ± 0.001
72a	25	67.726 ± 3.517	25	0.050 ± 0.004	25	0.005 ± 0.000	25	0.011 ± 0.004	25	0.026 ± 0.010
80a	19	5.223 ± 0.105	19	0.011 ± 0.004	19	0.003 ± 0.000	19	0.004 ± 0.001	19	0.004 ± 0.000
88a	6	0.002 ± 0.000	6	0.001 ± 0.000	6	0.001 ± 0.000	6	0.001 ± 0.000	6	0.003 ± 0.001
89a	28	>300	32	4.042 ± 0.378	32	0.015 ± 0.003	32	0.021 ± 0.004	32	0.010 ± 0.004
8a	11	3.010 ± 0.338	11	0.011 ± 0.001	11	0.004 ± 0.001	11	0.025 ± 0.007	11	0.018 ± 0.006
91a	21	21.189 ± 0.333	21	0.043 ± 0.005	21	0.009 ± 0.003	21	0.009 ± 0.001	21	0.018 ± 0.010
92a	22	>300	28	189.211 ± 19.031	28	0.062 ± 0.007	28	0.066 ± 0.010	28	0.050 ± 0.019
94a	17	0.853 ± 0.032	17	0.004 ± 0.001	17	0.004 ± 0.001	17	0.004 ± 0.001	17	0.008 ± 0.001
96a	13	>300	23	>300	27	0.222 ± 0.055	27	1.155 ± 0.178	23	0.166 ± 0.025

Table L.3

tdMCS ($\theta = 2$) modular product and MCS performance information for the Franco compound pairs.

pair	BK		CP		MaxCliqueSeq		RASCAL		kcombu	
	Size	Time	Size	Time	Size	Time	Size	Time	Size	Time
80a	19	0.020 ± 0.010	19	0.003 ± 0.001	19	0.002 ± 0.000	19	0.003 ± 0.001	19	0.005 ± 0.002
89a	32	0.394 ± 0.056	32	0.017 ± 0.001	32	0.009 ± 0.004	32	0.013 ± 0.004	32	0.011 ± 0.002
2a	34	2.098 ± 0.569	34	0.280 ± 0.027	34	0.009 ± 0.004	34	0.010 ± 0.001	34	0.018 ± 0.004
57a	23	2.042 ± 0.460	23	0.007 ± 0.001	23	0.010 ± 0.001	22	0.009 ± 0.002	23	0.011 ± 0.004
92a	28	13.655 ± 1.289	28	0.019 ± 0.004	28	0.023 ± 0.003	28	0.028 ± 0.004	28	0.025 ± 0.006
31a	23	2.058 ± 0.426	23	0.007 ± 0.002	23	0.009 ± 0.004	22	0.009 ± 0.002	23	0.019 ± 0.003
19a	15	0.021 ± 0.006	15	0.002 ± 0.001	15	0.001 ± 0.000	15	0.002 ± 0.001	15	0.007 ± 0.001
100a	32	28.311 ± 3.099	32	0.068 ± 0.013	32	0.037 ± 0.005	32	0.053 ± 0.013	14	0.076 ± 0.016
91a	21	0.062 ± 0.016	21	0.004 ± 0.001	21	0.003 ± 0.001	20	0.004 ± 0.000	21	0.013 ± 0.004
94a	17	0.012 ± 0.003	17	0.002 ± 0.001	17	0.002 ± 0.000	17	0.003 ± 0.001	17	0.013 ± 0.004
72a	23	0.075 ± 0.008	23	0.004 ± 0.002	23	0.004 ± 0.001	23	0.004 ± 0.001	23	0.011 ± 0.003
18a	15	0.029 ± 0.016	15	0.001 ± 0.000	15	0.002 ± 0.001	15	0.003 ± 0.001	15	0.011 ± 0.002
34a	12	0.005 ± 0.002	12	0.001 ± 0.000	12	0.001 ± 0.000	12	0.001 ± 0.000	12	0.004 ± 0.001
47a	19	0.089 ± 0.025	19	0.004 ± 0.002	19	0.005 ± 0.003	19	0.008 ± 0.001	19	0.027 ± 0.009
69a	16	0.097 ± 0.040	16	0.004 ± 0.001	16	0.004 ± 0.001	16	0.006 ± 0.003	16	0.025 ± 0.010
28a	8	0.009 ± 0.001	8	0.002 ± 0.001	8	0.002 ± 0.001	8	0.002 ± 0.001	8	0.006 ± 0.003
96a	26	27.602 ± 2.488	26	0.030 ± 0.006	26	0.033 ± 0.006	26	0.050 ± 0.006	26	0.145 ± 0.030
50a	10	0.016 ± 0.002	10	0.001 ± 0.001	10	0.001 ± 0.000	10	0.002 ± 0.001	10	0.005 ± 0.001
8a	10	0.031 ± 0.016	10	0.002 ± 0.001	10	0.003 ± 0.001	10	0.004 ± 0.001	7	0.007 ± 0.002
13a	7	0.009 ± 0.003	7	0.001 ± 0.000	7	0.002 ± 0.001	7	0.002 ± 0.001	7	0.005 ± 0.001
45a	6	0.005 ± 0.002	6	0.001 ± 0.000	6	0.001 ± 0.000	6	0.002 ± 0.001	6	0.005 ± 0.002
12a	8	0.042 ± 0.022	8	0.002 ± 0.001	8	0.003 ± 0.001	8	0.005 ± 0.001	8	0.012 ± 0.003
30a	6	0.014 ± 0.001	6	0.001 ± 0.000	6	0.002 ± 0.001	6	0.002 ± 0.001	6	0.006 ± 0.001
20a	13	2.397 ± 0.589	13	0.010 ± 0.004	13	0.012 ± 0.003	13	0.033 ± 0.007	11	0.025 ± 0.004
88a	5	0.001 ± 0.000	5	0.000 ± 0.000	5	0.001 ± 0.000	5	0.001 ± 0.000	5	0.001 ± 0.000

Table L.4

tdMCS ($\theta = 0$) modular product and *MCS* performance information for the Franco compound pairs.

pair	VFLibMCS		SMSD		fMCS		CDKMCS			BK	kcombu		CA_MCS	
	Size	Time	Size	Time	Size	Time	Size	Time	Size		Time	Size	Time	
80a	19	0.007 ± 0.004	19	0.007 ± 0.004	19	0.001 ± 0.000	19	0.002 ± 0.001	19	0.077 ± 0.009	19	0.004 ± 0.001	19	0.001 ± 0.000
89a	32	0.058 ± 0.027	32	0.441 ± 0.560	32	0.002 ± 0.000	32	0.011 ± 0.004	32	6.292 ± 0.308	32	0.008 ± 0.003	32	0.003 ± 0.001
2a	34	0.047 ± 0.018	34	0.067 ± 0.035	34	0.003 ± 0.001	34	0.008 ± 0.002	34	15.690 ± 2.118	34	0.012 ± 0.001	34	0.003 ± 0.001
57a	23	0.937 ± 0.664	23	0.030 ± 0.010	23	0.002 ± 0.001	23	0.011 ± 0.003	23	1.837 ± 0.310	23	0.025 ± 0.005	23	0.003 ± 0.001
92a	29	3.204 ± 1.791	29	3.632 ± 1.683	28	0.008 ± 0.003	28	0.060 ± 0.016	28	21.349 ± 2.430	28	0.053 ± 0.005	28	0.005 ± 0.001
31a	23	1.020 ± 0.769	23	0.806 ± 0.325	23	0.002 ± 0.001	23	0.012 ± 0.002	23	1.714 ± 0.195	23	0.021 ± 0.002	23	0.002 ± 0.000
19a	15	0.447 ± 0.067	15	0.443 ± 0.058	15	0.004 ± 0.002	15	0.002 ± 0.001	15	0.015 ± 0.006	15	0.008 ± 0.003	15	0.001 ± 0.000
100a	33	1.018 ± 0.355	33	2.925 ± 1.782	32	0.045 ± 0.008	16	0.158 ± 0.098	33	222.855 ± 8.435	33	0.097 ± 0.010	33	0.038 ± 0.007
91a	21	0.691 ± 0.384	21	0.600 ± 0.268	21	0.001 ± 0.000	21	0.005 ± 0.003	21	0.144 ± 0.024	21	0.030 ± 0.008	21	0.002 ± 0.000
94a	17	0.424 ± 0.082	17	0.497 ± 0.131	17	0.001 ± 0.000	17	0.003 ± 0.001	17	0.014 ± 0.003	16	0.005 ± 0.001	17	0.001 ± 0.000
72a	23	0.460 ± 0.079	23	1.448 ± 0.821	23	0.035 ± 0.013	23	0.035 ± 0.012	23	0.361 ± 0.021	23	0.013 ± 0.002	23	0.007 ± 0.001
18a	15	0.674 ± 0.388	15	0.454 ± 0.067	15	0.029 ± 0.007	15	0.003 ± 0.001	15	0.013 ± 0.003	15	0.007 ± 0.002	15	0.001 ± 0.000
34a	12	0.003 ± 0.000	12	0.003 ± 0.001	12	0.001 ± 0.000	12	0.001 ± 0.000	12	0.004 ± 0.001	12	0.004 ± 0.001	12	0.001 ± 0.000
47a	20	0.886 ± 0.651	20	0.947 ± 0.628	20	0.040 ± 0.008	20	0.050 ± 0.010	20	0.050 ± 0.012	18	0.015 ± 0.005	20	0.007 ± 0.001
69a	17	0.819 ± 0.420	17	1.016 ± 0.796	17	0.007 ± 0.002	17	0.035 ± 0.008	17	0.042 ± 0.011	17	0.021 ± 0.004	17	0.008 ± 0.000
28a	7	0.419 ± 0.062	7	0.459 ± 0.307	7	0.002 ± 0.001	7	0.004 ± 0.001	7	0.003 ± 0.001	7	0.005 ± 0.002	7	0.004 ± 0.001
96a	28	3.043 ± 2.061	28	2.555 ± 1.503	27	0.030 ± 0.005	26	0.110 ± 0.077	27	27.127 ± 5.218	27	0.128 ± 0.025	27	0.042 ± 0.005
50a	10	0.402 ± 0.047	10	0.418 ± 0.053	10	0.001 ± 0.000	10	0.002 ± 0.000	10	0.004 ± 0.001	8	0.004 ± 0.002	8	0.004 ± 0.001
8a	7	0.478 ± 0.053	7	0.896 ± 0.268	7	0.003 ± 0.001	7	0.005 ± 0.001	7	0.005 ± 0.001	7	0.008 ± 0.001	7	0.005 ± 0.002
13a	6	0.405 ± 0.032	6	0.404 ± 0.043	7	0.002 ± 0.000	7	0.002 ± 0.000	7	0.003 ± 0.000	7	0.004 ± 0.001	3	0.003 ± 0.000
45a	4	0.398 ± 0.031	4	0.404 ± 0.018	4	0.002 ± 0.001	4	0.002 ± 0.000	6	0.003 ± 0.001	4	0.003 ± 0.000	4	0.003 ± 0.001
12a	7	0.666 ± 0.281	7	0.697 ± 0.320	7	0.002 ± 0.000	7	0.040 ± 0.010	7	0.013 ± 0.005	7	0.015 ± 0.004	7	0.007 ± 0.000
30a	6	0.493 ± 0.156	6	0.486 ± 0.300	6	0.002 ± 0.001	6	0.003 ± 0.000	6	0.003 ± 0.000	6	0.007 ± 0.001	6	0.003 ± 0.000
20a	12	2.151 ± 1.193	12	2.528 ± 1.202	16	0.124 ± 0.027	16	0.772 ± 0.610	16	0.496 ± 0.104	14	0.037 ± 0.010	15	0.029 ± 0.010
88a	2	0.427 ± 0.064	2	0.438 ± 0.052	3	0.001 ± 0.001	3	0.000 ± 0.000	3	0.001 ± 0.000	3	0.001 ± 0.000	3	0.001 ± 0.000

Table L.5

mCSC performance information for the Franco compound pairs. CA_MCS refers to ChemAxon_MCS, and kcombu was used to calculate the cMCES.