A Polynomial-time Maximum Common Subgraph Algorithm for Outerplanar Graphs and its Application to Chemoinformatics

Leander Schietgat Jan Ramon Maurice Bruynooghe

Department of Computer Science - Katholieke Universiteit Leuven
`{firstname.lastname}@cs.kuleuven.be`

Corresponding Author

Leander Schietgat
Department of Computer Science - Katholieke Universiteit Leuven
Celestijnenlaan 200A
3001 Leuven
Belgium

Tel: +32 (0)16 32 76 37
Fax: +32 (0)16 32 79 96
E-mail: leander.schietgat@cs.kuleuven.be

Abstract

Metrics for structured data have received an increasing interest in the machine learning community. Graphs provide a natural representation for structured data, but a lot of operations on graphs are computationally intractable. In this article, we present a polynomial-time algorithm that computes a maximum common subgraph of two outerplanar graphs. The algorithm makes use of the block-and-bridge preserving subgraph isomorphism, which has significant efficiency benefits and is also motivated from a chemical perspective. We focus on the application of learning structure-activity relationships, where the task is to predict the chemical activity of molecules. We show how the algorithm can be used to construct a metric for structured data and we evaluate this metric and more generally also the block-and-bridge preserving matching operator on 60 molecular datasets, obtaining state-of-the-art results in terms of predictive performance and efficiency.

Keywords: metrics for structured data, graph mining, chemoinformatics, learning structure-activity relationships

1 Introduction

Metrics are important components of several machine learning methods, such as instance-based learning or clustering. Recently, there has been an increased interest in metrics that express a similarity between *structured* objects instead of propositional objects. This is relevant for multiple application domains including drug discovery [5], image recognition and computer vision [42].

The application on which we focus is the learning of structure-activity relationships (SAR) in drug discovery, which is an important step in the identification of molecules that play an active role in the regulation of biological processes or disease states. Even though it has become easier during the last years to acquire molecular data, this process, which is often referred to as screening, is still costly and time-consuming. For this reason, pharmaceutical companies are interested in SAR techniques that can predict the activity of molecules automatically and can therefore seriously decrease the amount of molecules that should be screened in the lab. Various properties of molecules are of interest in drug development, such as toxicity, carcinogenicity or biodegradability.

Since it is widely known that molecules with a similar structure tend to have the same function [24], structural similarity search among small molecules is the standard tool used for virtual screening and in silico drug development. The task then comes down to finding an appropriate similarity measure between molecules. Such a structural similarity measure should ideally fulfil two requirements: (1) it should be efficiently computable, which is important when analyzing large molecular databases, and (2) the notion of similarity should discriminate between molecules w.r.t. the activity of interest. Finding such similarity measures is one of the current challenges in chemoinformatics [5, 12].

Because graphs provide a natural representation of the atom-bond structure of molecules, they have become very popular in the context of SAR: each vertex of the graph then represents an atom, while an edge represents a molecular bond [38]. However, algorithms on graphs that aim at using all available structural information often involve the matching of subgraphs or other combinatorial operations trying to align graphs optimally. For this reason, typical approaches have resorted to a propositional representation by transforming molecules into vectors. In such transformations either information is lost or the size of the resulting representation can grow exponentially [12].

In order to define a similarity measure between molecules, we use the notion of a maximum common subgraph (MCS) between their graph representations. Intuitively, an MCS of two graphs is a connected subgraph common to both graphs, such that no other common subgraph has a size that is strictly larger. The ratio between the size of the MCS of the two graphs and their own size then reflects their similarity. The idea behind this is that an MCS of two molecular graphs may reflect shared properties that relate to the molecular activity. An example of an MCS of two molecular graphs is shown in Fig. 1. Given our target application, the motivation for this MCS-based similarity measure is threefold: (1) the MCS of structurally related drugs is likely to be an important component of their activities [4], (2) it is known that methods based on computing MCSs

Figure 1: A maximum common subgraph (MCS) of two molecular graphs (highlighted in gray).

work well for this type of tasks [38], (3) MCSs are able to reflect local similarities in the graphs rather than global ones, and (4) because of the visual nature of graphs, it is possible for chemists to investigate the MCSs that are predicted to be responsible for certain activities.

Finding the possibly multiple MCSs of two graphs, however, is known to be an NP-hard problem [14]. Nevertheless, previous theoretical work on graphs has shown that in many cases, when certain constraints on the structure of the graphs hold, efficient matching algorithms exist [33, 30]. Sequences, trees and outerplanar graphs (the latter are planar graphs which can be embedded in the plane in such a way that all of their vertices lie on the boundary of the outer face) are examples of such subclasses of general graphs. Investigation on the NCI¹ database, a collection of datasets containing over 250,000 chemical compounds, revealed that 94.5% of these graphs are outerplanar while only 8.8% of them are trees [21]. Hence, algorithms that can deal with outerplanar graphs have a much broader applicability than those handling trees, while still being efficient.

Recently, Horváth et al. [21, 22] introduced a new “block-and-bridge-preserving” (BBP) matching operator for outerplanar graphs. It is based on the idea that it only makes sense to match “similar” parts of the graphs. The authors show that, using this operator, the computation of pattern embeddings, that is, checking whether a pattern is subgraph isomorphic to graphs in the dataset, is polynomial, such that frequent patterns can be mined efficiently in outerplanar graphs.

In this research, the task is to find a connected graph of maximum size that is subgraph isomorphic to two given graphs. Maximum common subgraphs can be regarded as local patterns in the sense that they are frequent with support 2 (or more). It is shown in [40] that they are useful patterns for predictive tasks. In Schietgat et al. [39], we presented a polynomial algorithm that computes a maximum common subgraph between two outerplanar graphs under the block-and-bridge preserving subgraph isomorphism and we showed that a metric using

¹National Cancer Institute (<http://cactus.nci.nih.gov/>).

this algorithm obtains favorable results on several prediction tasks in chemoinformatics. This metric has three important properties: (1) it is computable in polynomial time, (2) as it reflects the size of the MCS, it has an intuitive meaning making results of methods such as instance-based learning interpretable, and (3) it provides the right level of abstraction in order to discriminate between molecules with different activities.

In this paper, we extend this work along several dimensions:

- We provide a more detailed technical description of the MCS algorithm, with a proof of its correctness. We also provide new complexity bounds on the algorithm and prove these bounds.
- We compare our algorithm to a state-of-the-art algorithm that computes MCSs under the general subgraph isomorphism, in terms of predictive performance and efficiency.
- We extend our experimental evaluation, not only showing that our metric outperforms state-of-the-art metrics on several classification tasks from chemoinformatics, but also by showing the advantages of the BBP subgraph isomorphism over the general subgraph isomorphism in terms of predictive performance and efficiency in other learning techniques.

The text is organized as follows. In Sect. 2, we start by introducing the necessary graph-theoretical concepts and then we present our MCS algorithm in detail. In Sect. 3, we present an experimental evaluation of the algorithm. In Sect. 4, we discuss previous work and in Sect. 5, we draw conclusions and present future work.

2 Computing a Maximum Common Subgraph of Two Outerplanar Graphs

In this section, we first give the necessary graph-theoretical concepts (Sect. 2.1) and then present the MCS algorithm (Sect. 2.2).

2.1 Graph-theoretical concepts

This section gives the relevant definitions necessary to understand the algorithm that computes an MCS of two outerplanar graphs. For an overview of graph theory, we refer to an introductory text book [13].

Definition 1 (Rooted Graph). *A labeled graph G is a quadruple (V, E, Σ, λ) , with V a finite set of vertices and $E \subseteq \{\{u, v\} \mid u, v \in V, u \neq v\}$ a set of edges. Σ is a finite set of labels and $\lambda : V \cup E \rightarrow \Sigma$ is a function assigning a label to each element of $V \cup E$. A rooted graph G is a quintuple $(V, E, \Sigma, \lambda, r)$ with (V, E, Σ, λ) a labeled graph and r , the root, one of its vertices.*

Given a graph G , we denote its vertices with $V(G)$ and its edges with $E(G)$. If the graph is rooted, we denote its root with $r(G)$. Sometimes, it will be convenient to denote a rooted graph as G^r with r the root of the graph. The set of all graphs is denoted as $\mathcal{G}$. As we define edges as sets of two distinct vertices, it means that our graphs do not have reflexive edges (loops) and that all edges are undirected, i.e., that we consider simple graphs.

Definition 2 (Size). *The size $|\cdot| : \mathcal{G} \rightarrow \mathbb{R}$ of a graph is a function mapping a graph to a real number of the form $|G| = \sum_{x \in V(G) \cup E(G)} w_{\lambda_G(x)}$, where each possible label of $l \in \Sigma$ has been assigned a non-negative weight w_l .*

Example 1. *Assume that all vertices of $v \in V(G)$ have the same label $\lambda_G(v) = l_1$ and all the edges $e \in E(G)$ have the same label $\lambda_G(e) = l_2$. Let $w_{l_1} = 1$ and $w_{l_2} = 0$. Then, the corresponding size function maps every graph G on the number of vertices of G .*

Definition 3 (Path, Cycle). *A sequence $x_0, x_1, \dots, x_n$ of vertices is a path from x_0 to x_n iff $\{x_i, x_{i+1}\} \in E(G)$ for all $i \in [0, n-1]$. A cycle $x_0, \dots, x_n$ is a path such that $x_0 = x_n$. A path without repeated vertices is a simple path²; a cycle without repeated vertices apart from the start and end vertex is a simple cycle.*

Definition 4 (Connected). *A graph G is connected if there is a path between any pair of its vertices in G ; it is biconnected if for any two vertices u and v of G , there is a simple cycle in G containing u and v .*

Definition 5 (Planar graph). *A graph is planar if it has a planar embedding i.e., it can be drawn in the plane in such a way that no two edges intersect except at a vertex in common. The regions formed by the edges in a planar embedding are called faces. There is one unbounded face, which is called the outer face.*

Definition 6 (Outerplanar graph). *An outerplanar graph is a planar graph which can be embedded in the plane in such a way that all of its vertices lie on the boundary of the outer face.*

We denote the set of all outerplanar graphs with $\mathcal{G}_{op}$. Note that the boundary of the outer face is not connected when the outerplanar graph is not connected.

Definition 7 (Tree). *A tree T is a graph for which there is a unique path between every pair of vertices $u, v \in T$.*

Definition 8 (Subgraph). *Let G and H be graphs. G is a subgraph of H , if (i) $V(G) \subseteq V(H)$, (ii) $E(G) \subseteq E(H)$, and (iii) $\lambda_G(x) = \lambda_H(x)$ holds for every $x \in V(G) \cup E(G)$.*

Definition 9 (Block). *A biconnected component or block of a graph G is a maximal subgraph of G that is biconnected.*

²In the literature, alternative terminology might be used. Our definition of a path is then called a walk, and our definition of a simple path is then called a path.

Definition 10 (Bridge). *In a graph G , a bridge is an edge of G that does not belong to a block.*

For a graph G we number the blocks and denote them $Bl_1(G)$, $Bl_2(G)$, $\dots$, $Bl_n(G)$. With $Bl(G)$, we denote all the edges that are involved in blocks of G and with $Br(G)$ all the bridges of G .

Any graph consists entirely of blocks and bridges, that is, its edges can be partitioned as: $E(G) = Br(G) \cup Bl_1(G) \cup \dots \cup Bl_n(G)$ with n the number of blocks, $Br(G) \cap Bl(G) = \emptyset$ and $\forall i, j$ ($i \neq j$) : $Bl_i(G) \cap Bl_j(G) = \emptyset$. Two blocks can share a vertex, but no edge. If they would share an edge, the blocks cannot be maximal biconnected subgraphs as their union would be larger and still biconnected.

In outerplanar graphs, the bridges are always adjacent to the outer face, while block edges are adjacent to two faces, one of which can possibly be the outer face.

Example 2. *Figure 2a shows an example of a non-outerplanar graph. Indeed, in all possible embeddings of the graph, there is one vertex (labeled v) that is not on the outside of the graph. The graphs in Fig. 2b and Fig. 2c are, however, outerplanar. Every vertex is labeled with a color representing a chemical element: black for carbon, white for hydrogen and blue for nitrogen. Note that the graph of Figure 2a, the edges between carbons and hydrogens are bridges, while the rest of the edges are involved in blocks. From a chemical viewpoint, blocks correspond to ring structures while bridges are linear fragments of the molecule.*

Definition 11 (Graph isomorphism). *Two graphs G and H are isomorphic under φ if φ is a function from $V(G)$ to $V(H)$ such that (i) φ is a bijection, (ii) $\forall u, v : \{u, v\} \in E(G) \Leftrightarrow \{\varphi(u), \varphi(v)\} \in E(H)$, (iii) $\forall u : \lambda_G(u) = \lambda_H(\varphi(u))$, (iv) $\forall u, v : \{u, v\} \in E(G) \Rightarrow \lambda_G(\{u, v\}) = \lambda_H(\{\varphi(u), \varphi(v)\})$.*

Definition 12 (Subgraph isomorphism). *A graph G is subgraph isomorphic to H under φ , denoted $G \preceq_\varphi H$, iff G is isomorphic to a subgraph of H under φ .*

If the function φ is clear from the context, we omit it from the notation, i.e., $G \preceq H$. The subgraph isomorphism problem, which decides whether G is subgraph isomorphic to H is known to be NP-complete [14]; this also holds for outerplanar graphs [45].

Definition 13 (BBP subgraph isomorphism). *A subgraph isomorphism from G to H under φ is a block-and-bridge-preserving (BBP) subgraph isomorphism if (i) $\forall u, v : \{u, v\} \in Br(G) \Leftrightarrow \{\varphi(u), \varphi(v)\} \in Br(H)$ and (ii) φ maps edges from different blocks on edges from different blocks.*

We denote that a graph G is BBP subgraph isomorphic to H under φ by $G \sqsubseteq_\varphi H$. Again, we omit the subscript φ if not needed. The BBP subgraph isomorphism maps each bridge of G to a bridge of H and each block of G to a part of a block of H . While the latter part is biconnected, it is not necessarily the whole block. Contrary to the subgraph isomorphism problem, the BBP

Figure 2: Examples of molecular graphs. The colors of the vertices correspond to their labels: black for carbon, white for hydrogen and blue for nitrogen. (a) Example of a non-outerplanar graph. For all of its plane embeddings, v never lies in the outer face. (b) Two molecules with a maximum common subgraph computed under the general subgraph isomorphism ($\text{MCS}_{\preceq}$), highlighted in gray. (c) The same two molecules with a maximum common subgraph computed under the BBP subgraph isomorphism ($\text{MCS}_{\sqsubseteq}$), highlighted in gray.

subgraph isomorphism problem is computable in polynomial time for outerplanar graphs [21]. For trees, which form a subset of outerplanar graphs (they are connected and block-free), the BBP subgraph isomorphism is equivalent to the subtree isomorphism [41].

Definition 14 (Maximum common subgraph). *A common subgraph I of two graphs G and H is a connected graph such that I is isomorphic to a subgraph of G and to a subgraph of H ($I \preceq G$ and $I \preceq H$); it is a maximum common subgraph (MCS) when in addition there exists no other common subgraph J , such that $\text{size}(I) < \text{size}(J)$.*

Note that we only consider connected MCSs. We use the abbreviation $\text{MCS}_{\preceq}$ to stress that we refer to a maximum common subgraph under the (general) subgraph isomorphism and $\text{MCS}_{\sqsubseteq}$ when it is under the BBP subgraph isomorphism. As a block can have subblocks, an $\text{MCS}_{\sqsubseteq}$ can include only a part of a block of the given graphs. Since the BBP subgraph isomorphism is a restricted version of the general subgraph isomorphism, an $\text{MCS}_{\sqsubseteq}$ is subgraph isomorphic to one of the $\text{MCS}_{\preceq}$. For example, the $\text{MCS}_{\sqsubseteq}$ of the graphs in Fig. 2c is subgraph isomorphic to the $\text{MCS}_{\preceq}$ of the graphs in Fig. 2b. In the worst case, two general graphs can have an exponential number of $\text{MCS}_{\sqsubseteq}$ and computing an $\text{MCS}_{\sqsubseteq}$ of two general graphs is NP-hard [14]. In Sect. 2.2.5, however, we will show that it is possible to compute an $\text{MCS}_{\sqsubseteq}$ of two *outerplanar* graphs in polynomial time by using the block-and-bridge-preserving (BBP) subgraph isomorphism [21]. From an application point of view, the BBP subgraph iso-

morphism is motivated from the fact that, in molecules, cyclic structures and linear fragments usually behave differently, and hence treating them separately might well be a good thing.

Figure 2 shows a comparison between the $\text{MCS}_{\preceq}$ (b) and the $\text{MCS}_{\sqsubseteq}$ (c). In both examples, the MCS is highlighted in gray. Note that one of the edges is a bridge in the upper graph, while it belongs to a block in the lower graph (marked with a * in both graphs) and hence, it cannot be mapped under the BBP subgraph isomorphism. Chemists consider it relevant not to map linear fragments with fragments that are part of a ring structure. From now on, we often use MCS to indicate an $\text{MCS}_{\sqsubseteq}$.

We end this section with recalling some definitions and results about matchings, as introduced in [34].

Definition 15 (Matching). *A matching f between sets A and B is a relation (i.e. a subset of $A \times B$) such that for all $(a_1, b_1), (a_2, b_2) \in f$: $a_1 = a_2$ iff $b_1 = b_2$, so each element of A is associated with at most one element of B and vice versa.*

Definition 16 (Weighted maximal matching). *The weighted maximal matching problem, also known as the assignment problem, is an optimisation problem where two sets A and B are given together with a weight function $w : A \times B \rightarrow \mathbb{R}$ and the task is to find a matching m between A and B such that $\sum_{(a,b) \in m} w(a,b)$ is maximal.*

Computing a weighted maximal matching can happen in $O((n+m) \cdot n^{1/2})$, with $n = |A| + |B|$ and m the number of pairs between A and B having a weight greater than zero [19].

2.2 Algorithm

For simplicity of exposition, in this section we construct an $\text{MCS}_{\sqsubseteq}$ for a pair of connected outerplanar graphs. The extension to non-connected graphs is straightforward. So, from here onwards, we assume graphs are connected.

To compute an $\text{MCS}_{\sqsubseteq}$ of two outerplanar graphs, we apply a dynamic programming strategy. The graphs are decomposed into subgraphs (parts) and the MCS is computed from the MCSs between the subgraphs. To come to an efficient algorithm, we should not consider all possible subgraphs, but have to judiciously design a decomposition that includes enough subgraphs to find an MCS, while at the same time avoiding redundancy. A first step is to consider rooted graphs and to compute their *rooted maximum common subgraph* (RMCS), which we can define as follows.

Definition 17 (Rooted maximum common subgraph). *S is a rooted common subgraph of rooted outerplanar graphs G and H iff S is a common subgraph of G and H , i.e., $\exists \varphi_G, \varphi_H : S \sqsubseteq_{\varphi_G} G$ and $S \sqsubseteq_{\varphi_H} H$, and moreover, $\varphi_G(r(S)) = r(G)$ and $\varphi_H(r(S)) = r(H)$. S is a rooted maximum common subgraph ($\text{RMCS}_{\sqsubseteq}$) iff there does not exist another rooted common subgraph T that is larger, i.e., for which $\text{size}(T) > \text{size}(S)$.*

Without loss of generality, in the rest of the section we can assume that S is a subgraph of G with the same root as G , i.e., that φ_G is the identity mapping.

We first describe the decomposition of a connected rooted outerplanar graph into parts (Sect. 2.2.1). Then, in Sect. 2.2.2 we define the parts to be used in our algorithm and prove some properties about them. In Sect. 2.2.3, we give the algorithm for computing an MCS. In Sect. 2.2.4, we prove its correctness and finally, in Sect. 2.2.5, we describe the time complexity of the algorithm.

2.2.1 Decomposition of a connected rooted outerplanar graph

The BBP subgraph isomorphism maps bridges to bridges and blocks to (parts of) blocks, hence we can say that an MCS of two graphs is *block-preserving*, a concept we formally define as:

Definition 18 (Block-preserving subgraph). *A subgraph S of G is a block-preserving subgraph (BPS) iff bridges of S are bridges in G .*

From the definition, it follows that blocks in S are also (parts of) blocks in G . We distinguish between *basic-root* BPSs and *compound-root* BPSs.

Definition 19 (Basic-root and compound-root BPS). *A rooted BPS S of a graph G is basic-root if either $r(S)$ is incident with at most one edge (a bridge) or all edges incident with $r(S)$ belong to the same block. In the other case, S is a compound-root BPS.*

Let G and H be connected graphs and assume S is one of their MCSs ($S \sqsubseteq_{\varphi_G} G$ – note that φ_G is the identity mapping – and $S \sqsubseteq_{\varphi_H} H$). Arbitrarily selecting one vertex of G as root turns G into G^r , a rooted BPS, subgraph of G . The idea behind our approach is to decompose G^r into rooted parts in such a way that one can be sure to obtain a part P^s with its root s an element of S and such that $S^s \sqsubseteq_{\varphi_G} P^s$. H is decomposed in a similar way such that one can be sure to obtain a part Q^t with $S^s \sqsubseteq_{\varphi_H} Q^t$ with $\varphi_H(s) = t$. Now S^s is the RMCS of P^s and Q^t and the decomposition is such that this RMCS can be easily constructed from the RMCSs between the parts of P^s and Q^t . To achieve this and obtain an RMCS computation in polynomial time, we have to carefully design a decomposition that includes enough parts while avoiding too much redundancy.

Selecting a vertex r as root in a connected outerplanar graph G turns the graph into a rooted BPS of G which can either be basic-root or compound-root. First we describe how to decompose a compound-root BPS, then we do the same for a basic-root BPS.

Definition 20 (Elementary parts of a compound-root BPS). *Let G^r be a compound-root BPS of a graph G .*

- *For each bridge $\{r, i\}$ in G^r , the connected subgraph P^r with root r that remains after removing all edges $\{r, j\}$ for $j \neq i$ is an elementary part of G^r .*

Figure 3: A compound-root BPS G^r (a) and its two elementary parts. The elementary part P_1^r (b) is obtained by removing all edges not belonging to the block in which r is involved. The elementary part P_2^r (c) is obtained by removing all edges adjacent to r except for the bridge $\{r, i\}$.

- For each block B in G^r that contains r , the connected subgraph P^r with root r that remains after removing all edges $\{r, i\}$ that do not belong to B is an elementary part of G^r .

Note that the elementary parts of a compound-root BPS of G are basic-root BPSs of G . Figure 3 shows two examples of elementary parts of a compound-root BPS. Both are basic-root BPS elementary parts.

To decompose a basic-root BPS, the idea is to remove an edge containing the root. This is straightforward when there is only one such edge, such as graph (a) in Fig. 4. However, it is more involved when the root is part of a block. We need some notation to make clear which edge(s) are removed from a block. Every block of an outerplanar graph G has a unique Hamiltonian cycle over its vertices [33]. Given a planar embedding of G , we can enumerate the block's vertices according to either the clockwise ($\curvearrowright$) or the counterclockwise ($\curvearrowleft$) orientation. Given an orientation and a root, the block's vertices are ordered by the Hamiltonian cycle $\{r, \dots, r\}$ following the orientation; we say $i < j$ when i comes before j ; we can refer to the first, the last, to the vertices and/or edges in an interval $[u, v]$, and so on. When removing vertices and edges of a block along the Hamiltonian cycle, the resulting subgraph is not necessarily a block-preserving subgraph. If not, we call it a *block-splitting subgraph*.

Definition 21 (Block-splitting subgraph). *Let G^r be a basic-root BPS with root r belonging to a block B ; let o be an orientation and let $\{r, \dots, f, \dots, r\}$ be the Hamiltonian cycle following orientation o of block B . With $f \neq r$, the graph obtained by removing all vertices and edges on the interval $[r, f]$ of the Hamiltonian cycle (except for r and f themselves) and removing zero or more edges*

$\{f, i\}$ not belonging to block B (as well as all edges adjacent to a removed vertex and parts no longer connected to the remaining segment) is a *block-splitting subgraph (BSS)* of G^r if it has no edge between f and r . We denote it as $o[f, r]$ and f is its root.

Note that the subgraph construction in the definition results in a BPS when, after removing the edges of the interval $[r, f]$, there is still an edge between f and r . This is for example the case in graph (c) of Fig. 4 when removing the vertices and edges in $\curvearrowright^b[i, r]$. For computing an MCS, there will be no need for considering subgraphs where more than one block is split. As for BPSs, we can distinguish two kinds of BSSs.

Definition 22 (Basic-root and compound-root BSS). *A BSS $o[f, r]$ of a BPS G^r is a basic-root BSS if f participates only in edges of the split block; otherwise it is a compound-root BSS.*

When relevant to make the distinction, we write $o^c[f, r]$ and $o^b[f, r]$ for respectively a compound-root and a basic-root BSS over the same interval. To construct an MCS we do not need all BSSs of a basic-root BPS. Also here we make a judicious selection.

Definition 23 (Elementary parts of a basic-root BPS). *Let G^r be a basic-root BPS of a graph G . If r belongs to a bridge $\{r, i\}$, then the BPS P^i with root i that remains after removing the bridge is the only elementary part of G^r .*

Otherwise, r belongs to a block B . Let o be an orientation and $\{r, 1, \dots, n, r\}$ ($n > 1$) be the Hamiltonian cycle following orientation o .

- *For each edge $\{r, i\}$ with $1 \leq i < n$, the BSS $o[i, r]$ obtained by removing $\{r, i\}$ and all vertices of B not belonging to the block interval $[i, r]$ (and the edges adjacent to such vertices and the parts not connected to the vertices on the interval) is an elementary part of G^r .*
- *In case vertex 1 belongs to more than two edges of B , it is part of a subblock not containing r . In that case, let $j \neq r$ be the largest vertex (in the order $\{1, \dots, n\}$ of orientation o) such that $\{1, j\}$ is an edge of B . The BPS P^1 with root 1 obtained by removing all vertices of B not belonging to the block interval $[1, j]$ (and the edges adjacent to such vertices and the parts not connected to the vertices on the interval $[1, j]$) is an elementary part of G^r .*

The elementary parts can be basic-root or compound-root, depending on the property of their root. An example is shown in Fig. 4. In graph (a), the root is on a bridge and there is a single BPS, namely graph (b). In graph (c), the root belongs to a block. There are two BSSs according to orientation $\curvearrowright$ (graphs (d) and (e)) and two according to orientation $\curvearrowleft$ (graphs (f) and (g)). Finally, for orientation $\curvearrowright$, there is also a BPS (graph (h)).

The attentive reader may wonder whether both orientations need to be considered. As will become clear, when computing the MCS of graphs G and H ,

Figure 4: (a) A basic-root BPS G^r . (b) Its only elementary part, a compound-root BPS. (c) Another basic-root BPS G^r , now with r belonging to a block. It has 5 elementary parts. Graphs (d), (e), (f) and (g) are basic-root BSSs, graph (h) a basic-root BPS.

a fixed orientation will suffice when decomposing G . However, for decomposing H , both orientations will be needed.

We still have to decompose a BSS into elementary parts. For a compound-root BSS, this is very similar as for a compound-root BPS. The parts almost partition the graph, only the root is shared between them.

Definition 24 (Elementary parts of a compound-root BSS). *Let $o^c[i, r]$ be a compound-root BSS of G^r .*

- *For each bridge $\{i, j\}$ in $o^c[i, r]$, the connected basic-root BPS P^i that remains after removing all edges $\{i, k\}$ for $k \neq j$ is an elementary part of $o^c[i, r]$.*
- *For each block B in $o^c[i, r]$ that contains i and is different from the split block, the connected basic-root BPS P^i that remains after removing all edges $\{i, j\}$ that do not belong to B is an elementary part of $o^c[i, r]$.*
- *The connected basic-root BSS $o^b[i, r]$ of G^r obtained by removing all edges*

Figure 5: (a) A compound-root BSS $o^c[i, r]$. Graphs (b) and (c) are its two BPS elementary parts; graph (d) is its BSS elementary part.

$\{i, j\}$ of $o^c[i, r]$ not belonging to the split block is an elementary part of $o^c[i, r]$.

An example is shown in Fig. 5. Graph (a) has three elementary parts, two BPS parts (graphs (b) and (c)) and one BSS part (graph (d)).

The decomposition of a basic-root BSS of a BPS G^r is similar to the decomposition of G^r itself.

Definition 25 (Elementary parts of a basic-root BSS). *Let $o^b[i, r]$ be a basic-root BSS of G^r . Let $\{i, j, \dots, n, r\}$ be what remains of the Hamiltonian cycle along orientation o of block B . If $i = n$ i.e., only one edge is left, then there are no elementary parts. Otherwise:*

- *For each edge $\{i, k\}$ with $i < k \leq n$, the connected BSS $o[k, r]$ obtained by removing $\{i, k\}$ and all vertices of B not belonging to the block interval $[k, r]$ is an elementary part of o^b . It is denoted as $o[k, r]$.*
- *In case vertex j belongs to more than two edges of B , it is part of a subblock not containing i . In that case let $k \neq r$ be the largest vertex such that $\{j, k\}$ is an edge of B . The connected BPS P^j with root j obtained by removing all vertices of B not belonging to the block interval $[j, k]$ is an elementary part of $o^b[i, r]$.*

An example is given in Fig. 6. The graph (a) has two BSS parts (graphs (b) and (c)) and one BPS part (graph (d)). In constructing $o[k, r]$, the vertex i is removed as well as the block's edges connected with vertex i ; note that there can be several such edges. Again, the elementary parts can be basic-root or compound-root, depending on the property of their root.

Figure 6: (a) A basic-root BSS $o[i, r]$. Graphs (b) and (c) are its BSS elementary parts; graph (d) is its BPS elementary part.

2.2.2 Parts and their properties

Having defined decompositions of rooted BPSs and rooted BSSs, we can now formalize our claims. As before, G and H are the given connected rooted out-epplanar graphs. First, we define which parts of G are used during the search for an MCS.

Definition 26 (Parts). *Given a graph G , let r be an arbitrarily selected vertex of G . Then $Parts(G^r)$ is inductively defined as follows:*

- $G^r \in Parts(G^r)$
- If $P^p \in Parts(G^r)$ then the elementary parts of P^p are in $Parts(G^r)$.

For every connected subgraph S of G , we can claim that $Parts(G^r)$ has an element P with its root in S and with S a subgraph of P .

Proposition 1. *Let S be a non-empty and connected subgraph that is block-and-bridge-preserving subgraph isomorphic to G . Then, $Parts(G^r)$ contains a rooted BPS P such that $r(P) \in S$ and S is a subgraph of P .*

Proof. Note that $Parts(G^r)$ contains G^r and that G^r is a BPS that contains S . If the root of G^r is part of S , then the proposition is satisfied. Otherwise, we will gradually reduce G^r , always selecting a part that contains S , until we obtain a part with the desired property. More formally, let G^r be R . If $r(R) \in S$, then R , being a BPS, satisfies the proposition. Otherwise, R satisfies the following invariant: R is a BPS or a BSS, S is a subgraph of R and $r(R) \notin S$. The

proposition follows from the argument made below that R has an elementary part R' (and hence a member of $\text{Parts}(G^r)$) that either satisfies the proposition or satisfies the invariant. As S is non-empty, the proposition then follows. The argument is by case analysis.

- When R is a compound-root BPS or a compound-root BSS, each of the elementary parts is either a basic-root BPS or a basic-root BSS and does not overlap with the others except for the root $r(R)$, while their union covers the whole graph. Because $r(R)$ is not part of S , S must be a subgraph of one of the elementary parts, say R' . This R' satisfies the invariant.
- When R is a basic-root BPS and $r(R)$ is incident with a bridge, its only elementary part R' is obtained by removing that bridge. As $r(R) \notin S$, S is a subgraph of R' ; either R' is the BPS that satisfies the proposition or R' satisfies the invariant.
- When R is a basic-root BPS and $r(R)$ is part of a block, R has one or more BSS elementary parts and eventually one BPS elementary part. We do a case analysis and use the notations of Definition 23.
 - When the BPS P^1 exists and node 1 belongs to S , then P^1 satisfies the proposition.
 - Otherwise, the BSS $o[1, r]$ satisfies the invariant.³
- When R is a basic-root BSS $o[i, r]$, it has one or more BSS elementary parts and eventually one BPS elementary part. We do a case analysis and use the notations of Definition 25.
 - If the BPS P^j exists and vertex j belongs to S , then P^j satisfies the proposition.
 - Otherwise, the BSS $o[j, r]$ satisfies the invariant.

□

For this proof, it suffices to use only one orientation when constructing the elementary parts of a basic-root BPS with a root belonging to a block.

Now, we turn our attention to the decomposition of H , where we consider every vertex as root.

Definition 27 (Parts^*). *Given a graph H , $\text{Parts}^*(H)$ is defined as follows:*

$$\text{Parts}^*(H) = \cup_{s \in V(H)} \text{Parts}(H^s)$$

Proposition 2. *Let S be a rooted connected graph such that $r(S) = s$ and $S \sqsubseteq_{\varphi_H} H$. Then $\text{Parts}^*(H)$ contains a rooted BPS P such that S is a subgraph of P and $\varphi_H(s) = r(P)$.*

³The other BSSs $o[i, r]$ are used in the proof of Proposition 3.

Algorithm 1 Computing an MCS of two outerplanar graphs G and H

Require: G and H are outerplanar graphs; $Pairs(G, H)$ as in Definition 28.

Ensure: $MCS_{best_candidate}$ is a subgraph of G and an $MCS_{\sqsubseteq}$ of G and H .

```

1:  $MCS_{best\_candidate} \leftarrow ((\emptyset, \emptyset), 0)$ 
2: for all  $(P, Q) \in Pairs(G, H)$ 
3:    $RMCS = RMCS(P, Q)$ 
4:   if  $P$  is a BPS and  $|RMCS| > |MCS_{best\_candidate}|$ 
5:      $MCS_{best\_candidate} \leftarrow RMCS$ 
6: return  $MCS_{best\_candidate}$ 
7: function  $RMCS(P, Q)$ 
8:   if  $P$  and  $Q$  are compound-root graphs
9:     return  $RMCS_{COMPOUND}(P, Q)$ 
10:  else if  $P$  and  $Q$  are basic-root BPS graphs
11:    return  $RMCS_{BPS}(P, Q)$ 
12:  else  $P$  and  $Q$  are basic-root BSS graphs
13:    return  $RMCS_{BSS}(P, Q)$ 

```

Proof. $H^{\varphi_H(s)}$ satisfies the claim and is in $Parts^*(H)$. □

Let S be an MCS of graphs G (mapping φ_G) and H (mapping φ_H); let r be an arbitrary vertex of G . We now have established that S has a vertex s such that $S^s \sqsubseteq_{\varphi_G} G^r$ (Proposition 1) and $S^s \sqsubseteq_{\varphi_H} H^{\varphi_H(s)}$ (Proposition 2), i.e., S^s is one of their RMCSs. This means that considering the subgraphs defined in $Parts(G^r)$ and $Parts^*(H)$ will lead to a valid RMCS between G^r and H^s .

2.2.3 Computing an MCS of two outerplanar graphs

The algorithm below constructs pairwise RMCSs for elements in $Parts(G^r)$ and $Parts^*(H)$. While doing so, it keeps track of the largest RMCS found so far; it uses a dynamic programming approach and stores results, so that the RMCS for each pair is computed only once. When all pairs have been processed, the largest RMCS found is an MCS of graphs G and H . To reduce the number of cases to four, it assumes that the initial graphs as well as the elementary parts of a basic-root graph are compound-root. If a compound-root graph turns out to be basic-root, the graph itself is its elementary basic-root part. So, $Parts(G)$ and $Parts^*(H)$ may have two copies of some graphs, one copy labeled as compound-root, the other as basic-root. It then suffices to compute the RMCS of pairs of basic-root graphs and of pairs of compound-root graphs. The pairs to be considered can be formally defined as follows.

Definition 28 ($Pairs(G, H)$). $Pairs(G, H)$ is the set of pairs $\{(P^p, Q^q) \mid \lambda(p) = \lambda(q), P^p \in Parts(G^r), Q^q \in Parts^*(H), P^p \text{ is basic-root (compound-root) iff } Q^q \text{ is basic-root (compound-root), and } P^p \text{ is a BPS (BSS) iff } Q^q \text{ is a BPS (BSS)}\}$.

Algorithm 1 shows the top level of the method. Maximum common subgraphs are represented by pairs consisting of a graph and a size. A graph in

Algorithm 2 Computing an RMCS of two compound-root graphs G^r and H^s

Require: G^r and H^s are compound-root graphs; their roots have the same label.

```

1: function RMCSCompound( $G^r, H^s$ )
2:    $M \leftarrow \text{MAXMATCH}(\text{EPS}(G^r), \text{EPS}(H^s))$ 
3:    $\text{RMCS} \leftarrow \cup_{(X,Y) \in M} \text{GRAPH}(\text{RMCS}(X, Y))$ 
4:    $\text{size} \leftarrow \sum_{(X,Y) \in M} |\text{RMCS}(X, Y)| - (|M| - 1)w(\lambda(r))$ 
5:   return ( $\text{RMCS}, \text{size}$ )

```

turn is represented as a pair (V, E) with V the vertices and E the edges. We use the functions `GRAPH`, `EDGES`, `VERTICES` and $|\cdot|$ to select respectively the graph, the edges, the vertices and the size of an MCS. We need not to specify the labels of an MCS as the MCS is a subgraph of G and it inherits the labels of G . We will write $(V_1, E_1) \cup (V_2, E_2)$ as a short notation for $(V_1 \cup V_2, E_1 \cup E_2)$. A global variable $\text{MCS}_{\text{best_candidate}}$ keeps track of a maximum common subgraph that has been found so far; it is initialized as the empty graph (size 0). The algorithm iterates over all pairs (P, Q) of rooted graphs. The RMCS is computed by the function `RMCS` which does a case analysis, and, depending on the case (compound-root graphs, basic-root BPS or basic-root BSS), calls one of `RMCSCompound`, `RMCSBPS` or `RMCSBSS`. The results of the function `RMCS(P, Q)` are stored so that the function is evaluated only once for each pair. The value of $\text{MCS}_{\text{best_candidate}}$ is updated when the RMCS of a pair of BPSs was computed and the found RMCS is larger than the currently best common subgraph (under BBP subgraph isomorphism).

First, we discuss the function `RMCSCompound` (Algorithm 2) that computes an RMCS of two compound-root graphs G^r and H^s with the same label in their roots. A compound-root graph has basic-root graphs as elementary parts. An RMCS is obtained by computing a weighted maximal matching between pairs of elementary parts, using the size of the RMCS as weight (see Proposition 3 in the next section). The RMCS is then the union of the RMCSs of the pairs that participate in the maximal matching. The algorithm uses `MAXMATCH` to compute a maximal matching and the function `EPS` to access the sets of elementary parts. These functions are omitted. When computing the size of the RMCS, simply adding the sizes of the RMCSs of the pairs participating in the matching would count $w(\lambda(r))$, the weight of the label of the root, $|M|$ times, so a correction is made.

Second, we discuss the RMCS computation of two basic-root BPS graphs (Algorithm 3). It first handles the case that the root belongs to a bridge. If so, both graphs have a single elementary part (accessed with the function `EP`). If the two bridges match, then the RMCS consists of the bridge and the RMCS between the elementary parts; otherwise, the RMCS consists of the root. If the root belongs to a block in one graph and to a bridge in the other, then the RMCS consists of the root. This is the initial RMCS and is returned as the for loop is skipped in that case. If the root belongs to a block in both graphs then the algorithm has to consider the pairs of elementary parts that are BSSs. If

Algorithm 3 Computing an RMCS of two basic-root BPSs G^r and H^s

Require: G^r and H^s are two basic-root block-preserving graphs; $\lambda(r) = \lambda(s)$.

```

1: function RMCSBPS( $G^r, H^s$ )
2:   if  $r$  is part of the bridge  $\{r, r'\}$  and  $s$  part of the bridge  $\{s, s'\}$ 
3:     if  $\lambda(\{r, r'\}) = \lambda(\{s, s'\}) \wedge \lambda(r') = \lambda(s')$  // the bridges match
4:        $RMCS \leftarrow \text{GRAPH}(RMCS(\text{EP}(G^r), \text{EP}(H^s))) \cup (\{r\}, \{\{r, r'\}\})$ 
5:        $size \leftarrow |RMCS(\text{EP}(G^r), \text{EP}(H^s))| + w(\lambda(\{r, r'\})) + w(\lambda(r))$ 
6:     else
7:        $RMCS \leftarrow (\{r\}, \emptyset)$ 
8:        $size \leftarrow w(\lambda(r))$ 
9:     return ( $RMCS, size$ )
10:  else
11:     $RMCS \leftarrow ((\{r\}, \emptyset), w(\lambda(r)))$ 
12:    for all pairs  $(o_G[i, r], o_H[i', s])$  such that  $o_G[i, r] \in \text{EPS}(G^r)$ ,
       $o_H[i', s] \in \text{EPS}(H^s)$  // pairs of BSSs
       $\wedge \lambda(\{i, r\}) = \lambda(\{i', s\}) \wedge \lambda(i) = \lambda(i')$  // and the edges match
13:      if  $|RMCS(o_G[i, r], o_H[i', s])| > 0$  // the pair has an RMCS
14:         $Cand \leftarrow \text{GRAPH}(RMCS(o_G[i, r], o_H[i', s])) \cup (\emptyset, \{\{r, i\}\})$ 
15:         $size \leftarrow |RMCS(o_G[i, r], o_H[i', s])| + w(\lambda(\{r, i\}))$ 
16:        if  $size > |RMCS|$ 
17:           $RMCS \leftarrow (Cand, size)$ 
18:  return  $RMCS$ 

```

the removed edges match and the pair has a non-empty RMCS, then a BPS is composed. If it improves the best result so far, it becomes the new RMCS.

Finally, Algorithm 4 computes the RMCS between two basic-root BSSs. Although it is quite similar to the previous, there are some differences. First, the root is known to be part of a block so the bridge case can be skipped. Second, it is possible that one of $\{r', r\}$ and $\{s', s\}$ is the final edge of the Hamiltonian path, in which case that one has no children. If there is an edge between r and r' and an edge with the same label between s and s' , then we have an RMCS that consists of this edge. Otherwise, we have to iterate over the pairs $o_G[r'', r]$ and $o_H[s'', s]$ of BSSs that are their elementary parts. If the edges $\{r'', r'\}$ and $\{s'', s'\}$ match and the pair has a non-empty RMCS, the RMCS between $o_G[r'', r]$ and $o_H[s'', s]$ can be extended into an RMCS of $o_G[r', r]$ and $o_H[s', s]$. Note that r' may have other edges $\{r', i\}$ with i in the interval $[r'', r]$. If i is in the RMCS (its image in $o_H[s'', s]$ is $\varphi_H(i)$) and the edge $\{s', \varphi_H(i)\}$ is present in $o_H[s', s]$ and matches with $\{r', i\}$, then the edge is added to the RMCS of $o_G[r', r]$ and $o_H[s', s]$ (they are collected in $Edges$). Note also, in case the RMCS of $o_G[r'', r]$ and $o_H[s'', s]$ is empty, but the edges $\{r, r'\}$ and $\{s, s'\}$ exist and match, then these edges form the RMCS between $o_G[r', r]$ and $o_H[s', s]$.

Algorithm 4 Computing an RMCS of two basic-root BSSs $o_G[r', r]$ and $o_H[s', s]$

Require: $o_G[r', r]$ and $o_H[s', s]$ are basic-root block-splitting graphs; $\lambda(r) = \lambda(s)$ and $\lambda(r') = \lambda(s')$.

```

1: function RMCSBSS( $o_G[r', r], o_H[s', s]$ )
2:   if one of  $o_G[r', r]$  and  $o_H[s', s]$  has no elementary parts
3:     if  $\lambda(\{r, r'\}) = \lambda(\{s, s'\})$  // the edges exist and match
4:       return  $((\{r', r\}, \{\{r', r\}\}), w(\lambda(r)) + w(\lambda(r')) + w(\lambda(\{r', r\})))$ 
5:     else
6:       return  $((\emptyset, \emptyset), 0)$ 
7:   else  $RMCS \leftarrow ((\emptyset, \emptyset), 0)$ 
8:   for all pairs  $(o_G[r'', r], o_H[s'', s])$  such that  $o_G[r'', r] \in \text{EPS}(o_G[r', r])$ 
      and  $o_H[s'', s] \in \text{EPS}(o_H[s', s])$  // BSSs
       $\wedge \lambda(\{r'', r'\}) = \lambda(\{s'', s'\}) \wedge \lambda(r'') = \lambda(s'')$  // the edges match
9:     if  $|RMCS(o_G[r'', r], o_H[s'', s])| > 0$  // the pair has an RMCS
10:       $Edges \leftarrow \{\{r', i\} | i \neq r'' \wedge i \in \text{VERTICES}(RMCS(o_G[r'', r], o_H[s'', s])) \wedge \{s', \varphi_H(i)\} \in o_H[s', s] \wedge \lambda(\{r', i\}) = \lambda(\{s', \varphi_H(i)\})\}$ 
11:       $Cand \leftarrow \text{GRAPH}(RMCS(o_G[r'', r], o_H[s'', s])) \cup (\{r'\}, \{\{r', r''\}\}) \cup (\emptyset, Edges)$ 
12:       $size \leftarrow |RMCS(o_G[r'', r], o_H[s'', s])| + w(\lambda(\{r', r''\})) + w(\lambda(r')) + \sum_{e \in Edges} w(\lambda(e))$ 
13:    else if  $\lambda(\{r, r'\}) = \lambda(\{s, s'\})$  // the edges exist and match
14:       $Cand \leftarrow (\{r', r\}, \{\{r', r\}\})$ 
15:       $size \leftarrow w(\lambda(r)) + w(\lambda(r')) + w(\lambda(r', r))$ 
16:    else
17:       $Cand \leftarrow (\emptyset, \emptyset); size \leftarrow 0$ 
18:    if  $size > |RMCS|$ 
19:       $RMCS \leftarrow (Cand, size)$ 
20: return  $RMCS$ 

```

2.2.4 Proof of correctness

In this section, we prove that the algorithms described in Sect. 2.2.3 work correctly. We do this by the following proposition, which shows that an RMCS between an element of $Parts(G^r)$ and an element of $Parts^*(H)$ is constructed from the RMCSs between their parts (Algorithms 2-4). This ensures us that the RMCS between G^r and H^s will be computed correctly.

Proposition 3. *Let P^p be a BPS (BSS) element of $Parts(G^r)$, Q^q a BPS (BSS) element of $Parts^*(H)$ and S^p be a BPS (BSS) that is an RMCS of P^p and Q^q , i.e., $S^p \sqsubseteq_{\varphi_G} P^p$ and $S^p \sqsubseteq_{\varphi_H} Q^q$. According to the different cases, Algorithms 2-4 compute an RMCS of P^p and Q^q that has the same size as S^p , which is constructed from the RMCSs between the elementary parts of P^p and Q^q .*

Proof. We prove this result by induction over the size of the RMCS S^p . When

the RMCS of P^p and Q^q is empty, i.e., p and q have a different label, then nothing is to be constructed. For the induction step, we do a case analysis.

- P^p , Q^q and S^p are BPSs.
 - **P^p , Q^q and S^p are compound-root BPSs** (Alg. 2) Definition 20 allows us to make the following observations: S^p is the union of its elementary parts.⁴ For each elementary part S_e^p of S^p , there is an elementary part P_e^p of P^p , such that $S_e^p \sqsubseteq_{\varphi_G} P_e^p$ and an elementary part Q_e^q of Q^q , such that $S_e^p \sqsubseteq_{\varphi_H} Q_e^q$. Moreover, we can claim that S_e^p is an RMCS of P_e^p and Q_e^q (if P_e^p and Q_e^q had a larger RMCS, then P^p and Q^q would have an RMCS larger than S^p). S_e^p is smaller than S^p , so we can apply the induction hypothesis and hence P_e^p and Q_e^q have an RMCS of size S_e^p . So, when computing the weighted maximal matching between the elementary parts of P^p and Q^q (using the size of their RMCS as weight), we obtain a matching which has as weight the sum of the sizes of the elementary parts of S^p . Taking the union of the RMCSs involved in this matching, we obtain an RMCS with a size equal to the size of S^p . Note that the matching cannot result in a larger weight as that would lead to an RMCS larger than S^p .
 - **P^p , Q^q and S^p are basic-root BPSs** (Alg. 3) Let o be the orientation that is chosen in the decomposition of P^p . Let $\{p, i, \dots, m, p\}$ be the Hamiltonian cycle following orientation o in S^p . S^p can be reconstructed from the edge $\{p, i\}$ and from $o_S[i, \dots, p]$, an elementary part of S^p (Definition 23). It follows from the same definition that $o_G[i, \dots, p]$ with $o_G = o_S$ is an elementary part of P^p and that Q^q has an elementary part $o_H[\varphi_H(i), \dots, \varphi_H(p)]$ (with o_H one of $\{\curvearrowright, \curvearrowleft\}$). Moreover, $o_S[i, \dots, p]$ is an RMCS of $o_G[i, \dots, p]$ and $o_H[\varphi_H(i), \dots, \varphi_H(p)]$, hence the induction hypothesis is applicable and both $o_G[i, \dots, p]$ and $o_H[\varphi_H(i), \dots, \varphi_H(p)]$ have an RMCS with the size of $o_G[i, \dots, p]$. Extending the latter RMCS with the edge $\{p, i\}$ gives an RMCS of P^p and Q^q which has the same size as S^p . As above, the assumption that $o_G[i, \dots, p]$ and $o_H[\varphi_H(i), \dots, \varphi_H(p)]$ have an RMCS larger than $o_G[i, \dots, p]$ contradicts that S^p is an RMCS of P^p and Q^q .
 - **S^p is a basic-root BPS and P^p and/or Q^q are compound-root BPSs** (Alg. 3) If P^p is compound-root, then it has an elementary part P'^p such that $S^p \sqsubseteq_{\varphi_G} P'^p$; if not, let $P'^p = P^p$. Similarly, we construct Q'^q such that $(S^p \sqsubseteq_{\varphi_H} Q'^q)$. Now we can apply the argument of the previous case to claim that P'^p and Q'^q have an RMCS of the size of S^p . This must also be an RMCS of P^p and Q^q as they cannot have a larger one.

⁴If the compound S^p has only itself as elementary basic BPS, there is strictly speaking no size reduction. However, note that one could define an auxiliary function $size'(G) = 2 * size(G) + 1$ if G is compound and $2 * size(G)$ otherwise. Formally, the induction is on $size'$.

- **P^p , Q^q and S^p are BSSs** Using Definition 24 and Definition 25, we can make a similar argument as for BPSs. Note however that for the case of basic-root BSSs (Alg. 4), the difference between $o_S[i, j, \dots, p]$ and its elementary part $o_S[j, \dots, p]$ consists of the vertex i , the edge $\{i, j\}$, and zero or more edges $\{i, k\}$. So when extending the RMCS between $o_G[j, \dots, p]$ and $o_H[\varphi_H(j), \dots, \varphi_H(p)]$, not only vertex i and edge $\{i, j\}$ have to be added, but also the edges $\{i, k\}$ for which k belongs to the RMCS of $o_G[j, \dots, p]$ and $o_H[\varphi_H(j), \dots, \varphi_H(p)]$.

□

The following theorem now directly follows from Propositions 1, 2 and 3.

Theorem 1. *Algorithm 1 computes an $MCS_{\sqsubseteq}$ of two given outerplanar graphs.*

2.2.5 Time complexity

In this section, we outline the proof for the following theorem:

Theorem 2. *Algorithm 1, computing an MCS under BBP subgraph isomorphism of two outerplanar graphs G and H , runs in time $O(|V(G)| \cdot |V(H)| \cdot (|V(G)| + |V(H)|)^{1/2})$.*

Proof. The result follows from counting the total time spent in Algorithms 1-4 and making use of known bounds on the running times of the algorithms used in the dynamic programming step.

For convenience, we account for the time spent in Alg. 1, which calls the appropriate function for every pair $(P, Q) \in Pairs(G, H)$, in the discussion of the functions themselves. More specifically, the iteration over the appropriate pairs is taken into consideration during the discussion of the three functions RMCSCompound, RMCSBPS and RMCSBSS.

First, the time needed to decompose G , i.e., to compute $Parts(G^r)$, can be bound by $O(|V(G)|)$. Since this is clearly not the dominating term in the complexity of the algorithm, we will only provide a sketch. The number of elementary parts of a BPS or BSS with root r depends on the degree of r . Therefore, the number of elements in $Parts(G^r)$ is linear to the number of edges in G . For outerplanar graphs it holds that $O(E(G)) = O(V(G))$ [33], so this gives us the above upper bound. In addition, the time needed to compute $Parts^*(H)$ can be bound by $O(|V(H)| \cdot |V(H)|)$. The time to decompose both G and H is then bound by $O(|V(G)| + |V(H)|^2)$. With $|V(H)| \leq |V(G)|$, this can be reduced to $O(|V(G)| \cdot |V(H)|)$.

Second, the total time spent in RMCSCompound (Alg. 2) can be bound by $O(|V(G)| \cdot |V(H)| \cdot (|V(G)| + |V(H)|)^{1/2})$. This can be explained as follows. Every vertex $g \in V(G)$ and every vertex $h \in V(H)$ has at most $deg(g)$ (resp. $deg(h)$) elementary parts involved in a maximal matching, with $deg(g)$ the degree of vertex g and $deg(h)$ the degree of vertex h . According to Hopcroft [19], the weighted maximal matching problem can be solved in time $O((n + m) \cdot n^{1/2})$, with n the number of nodes and m the number of edges in a bipartite graph.

Since $n = \deg(g) + \deg(h)$ and $m = \deg(g) \cdot \deg(h)$, this allows us to bound the time spent in RMCSCompound by T_{comp} :

$$T_{comp} = \sum_{g \in V(G)} \sum_{h \in V(H)} ((\deg(g) + \deg(h)) + (\deg(g) \cdot \deg(h))) \cdot (\deg(g) + \deg(h))^{1/2}.$$

In the non-trivial case, $\deg(g), \deg(h) \geq 1$, while $((\deg(g) + \deg(h)) + (\deg(g) \cdot \deg(h))) \leq 3((\deg(g) \cdot \deg(h)))$, $\sum_{v \in V(G)} \deg(v) = 2|E(G)|$ and for outerplanar graphs holds that $O(E(G)) = O(V(G))$ [33]. This gives us an upper bound on T_{comp} :

$$T_{comp} \leq \sum_{g \in V(G)} \sum_{h \in V(H)} 3 \cdot (\deg(g) \cdot \deg(h)) \cdot (\deg(g) + \deg(h))^{1/2}$$

$$T_{comp} = O(|V(G)| \cdot |V(H)| \cdot (|V(G)| + |V(H)|)^{1/2}).$$

Third, the total time spent in RMCSBPS (Alg. 3) can be bound by $O(|V(G)| \cdot |V(H)|)$. If $r \in V(G)$ and $s \in V(H)$ are part of a bridge, this function only requires a constant time. If r and s are involved in a block of G or H , we count the number of elementary parts of type BSS of the BPSs having r and s as roots, which can be upper bound by $\deg(r)$ and $\deg(s)$, respectively. Making use again of $\sum_{v \in V(G)} \deg(v) = 2|E(G)|$, this allows us to bound the time spent in RMCSBPS by T_{bps} :

$$\begin{aligned} T_{bps} &= \sum_{r \in V(G)} \sum_{s \in V(H)} \deg(r) \cdot \deg(s) = \sum_{r \in V(G)} \deg(r) \cdot \sum_{s \in V(H)} \deg(s) \\ &= O(|V(G)| \cdot |V(H)|). \end{aligned}$$

Finally, the time spent in RMCSBSS (Alg. 4) can be bound by $O(|V(G)| \cdot |V(H)|)$ as well. If $o_G[r', r]$ ($r, r' \in V(G)$) and $o_H[s', s]$ ($s, s' \in V(H)$) have no elementary parts, this function requires a constant time. Otherwise, all combinations of $\deg(r)$ elementary parts of $o_G[r', r]$ and $\deg(s)$ elementary parts of $o_H[s', s]$ need to be considered. As such, we can bound the time spent in RMCSBSS by T_{bss} :

$$\begin{aligned} T_{bss} &= \sum_{r \in V(G)} \sum_{s \in V(H)} \deg(r) \cdot \deg(s) = \sum_{r \in V(G)} \deg(r) \cdot \sum_{s \in V(H)} \deg(s) \\ &= O(|V(G)| \cdot |V(H)|). \end{aligned}$$

The overall time complexity is dominated by the time spent in RMCSCompound, so we can conclude that there is an algorithm running in a time bounded by a polynomial of degree 3, given the size of the vertices of the two graphs. $\square$

We end this section with a practical remark. In fact, we need not to consider all vertices of H as root. If we would only be dealing with tree structures, it would be sufficient to consider only their leaves as root. As outerplanar graphs also have blocks, we have to consider a larger class, which we call *root-candidates*.

Definition 29 (Root-candidate of an outerplanar graph). *A vertex is a root-candidate of an outerplanar graph if either it is incident with a single bridge or it belongs to a block.*

To decompose G , we select an arbitrary root-candidate r of G and decompose G^r . To decompose H , we consider all root-candidates s of H and decompose each H^s .

From now on, we refer to our algorithm as MCS-BBP. It can be downloaded at <http://dtai.cs.kuleuven.be/PMCSFG>.

3 Experimental Evaluation

In this section, we compare MCS-BBP against the state-of-the-art algorithms in terms of efficiency and predictive performance. We also want to investigate the advantages of using the BBP subgraph isomorphism as opposed to the general subgraph isomorphism. More precisely, we want to answer the following experimental questions:

- Q1** What are the differences in terms of efficiency and predictive performance between MCS-BBP and a state-of-the-art MCS algorithm that uses the general subgraph isomorphism?
- Q2** How does the predictive performance of a metric based on MCS-BBP compare to the predictive performance of state-of-the-art metrics?
- Q3** What is the influence on predictive performance when using the BBP subgraph isomorphism as matching operator in other learning techniques?

In order to answer Q1, we compare MCS-BBP against the MCS algorithm of Cao et al. [4]. To compare the predictive performance, we construct a metric from the MCS algorithms and evaluate them based on the performance of the metric in an instance-based learning (IBL) setting.

In order to answer Q2, we compare our metric based on the MCS $_{\square}$ to a metric based on 2D fingerprints [48] and a metric based on the WDK kernel [5]. These metrics are also evaluated through instance-based learning.

Although the gains in efficiency that can be obtained by using the BBP matching operator have been studied in the context of frequent subgraph mining by Horváth et al. [21, 22], the impact of the BBP matching operator on predictive performance has not been investigated before. Evaluating the BBP matching operator is interesting in its own and these results give us an answer to Q3. To this end, we compare the different matching operators when used in support vector machines (SVMs) [23], a method that has become very popular also for the classification of molecules (e.g., [44, 12, 5]). Rather than obtaining state-of-the-art results, the purpose in this context is investigating whether the BBP subgraph isomorphism has a larger predictive power compared to the general one in classification tasks from chemoinformatics.

3.1 Datasets

The Developmental Therapeutics Program (DTP) at the U.S. National Cancer Institute (NCI) has checked a large number of compounds for evidence of the ability to inhibit the growth of human tumor cell lines.⁵ The target values correspond to the parameter GI_{50} , the concentration that causes 50% growth inhibition.

In total, there are 60 datasets, each corresponding to a particular cell line. We use the roughly balanced datasets of [44], which have become a popular benchmark for QSAR algorithms, and are often referred to as NCI60. Each cell line has inhibition data on about 3500 compounds, which defines a binary classification problem. There are 3910 distinct molecules over all datasets. The average number of vertices is 23, while the average number of edges is 25.

Each molecule is described in the Tripos Sybyl MOL2 format⁶, from which we extract a graph in which the vertices are labeled with general atom types (e.g., N, C) and the edges are labeled single, double, triple, amide or aromatic. Hydrogen atoms are dropped.

In order to compare with the BBP-based methods, we have removed the non-outerplanar examples ($\approx 9.69\%$) from these datasets.⁷

3.2 Method

Q1: What are the differences between MCS-BBP and a state-of-the-art MCS algorithm that uses the general subgraph isomorphism in terms of efficiency and predictive performance? The algorithm of Cao et al. [4] computes an unconnected MCS under the general subgraph isomorphism and is not bounded by the restriction to outerplanar graphs. It makes use of heuristics to drastically reduce the search space compared to other MCS algorithms (see Sect. 4.1). To the best of our knowledge, this is the only MCS algorithm that is freely available in the public domain. The algorithm, part of a package called CHEMMINER, can be downloaded at <http://biowb.ucr.edu/ChemMineV2/help/mcs.html>. From now on, we refer to this algorithm as MCS-CAO.

To compare the efficiency, we selected a set of 50 graphs from NCI60 randomly, for which there are 1225 possible combinations of two graphs. For each of them, we run MCS-BBP (computing an $MCS_{\sqsubseteq}$) and MCS-CAO (computing an $MCS_{\preceq}$) and measure runtimes.

All experiments were run on an Intel Core2Quad 2.33 GHz processor. A time-out of 24 hours per MCS computation was enforced.

To compare the predictive performance of MCS-BBP and MCS-CAO, we construct a metric from the MCS. Bunke and Shearer [3] proposed a distance

⁵http://dtp.nci.nih.gov/docs/cancer/cancer_data.html

⁶<http://www.tripos.com/data/support/mol2.pdf>

⁷These examples cannot be handled efficiently and could be covered by an algorithm using the general subgraph isomorphism. In [40], an efficient feature generation approach based on the $MCS_{\sqsubseteq}$ was introduced, also covering the non-outerplanar examples.

function on graphs based on the maximum common subgraph:

$$d_{bs}(G, H) = 1 - \frac{|MCS(G, H)|}{\max(|G|, |H|)}, \quad (1)$$

with $|G|$ equal to the number of vertices in G . They proved that d_{bs} is a metric. We can easily extend this proof to the more general case where $|G|$ is determined by the function *size*. Some variants with similar properties and performances are reviewed in [37].

We instantiate the *size* of a graph as the sum of its number of vertices and its number of edges, that is, we choose $w_{\lambda_G(x)} = 1$ for every $x \in V(G) \cup E(G)$. We call the metric computed under the BBP subgraph isomorphism $d_{\sqsubseteq}$ and the metric computed under the general subgraph isomorphism $d_{\preceq}$.

To compare the predictive performance of the different metrics, we use k -nearest neighbour classification (kNN): in order to classify a given molecule, we select the k neighbour molecules that are closest according to the metric (we choose $k = 11$, which resulted in an optimal AUROC for all metrics). For each molecule, we obtain a prediction equal to the percentage of positive votes of its neighbors. In this way, we can rank the predictions and compute the area under the ROC curve (AUROC). We use leave-one-out cross-validation.

Q2: How does the predictive performance of a metric based on the MCS algorithm using the BBP subgraph isomorphism compare to the predictive performance of state-of-the-art metrics? Next to the MCS-based metrics, we construct a metric based on 2D fingerprints and a metric based on the WDK kernel. For each molecule, we construct a 1024 FP2-fingerprint using OpenBabel v2.1.1⁸ and we define a metric on these fingerprints using the Tanimoto coefficient, which is considered among chemists to produce state-of-the-art results for virtual screening [48]. We call this metric FP2. Although the WDK kernel is not intended to be used in this way, for reasons of comparison we define a metric according to the following formula: $d^2(x, y) = \kappa(x, x) - 2\kappa(x, y) + \kappa(y, y)$, with κ the WDK kernel function. We call this metric WDK.

Q3: What is the influence on predictive performance when using the BBP subgraph isomorphism as matching operator in other learning techniques? To test the performance of the BBP subgraph isomorphism in a broader context, we perform a propositionalization approach and transform molecules into bit-vectors using two approaches. The first approach mines frequent subgraphs under the BBP subgraph isomorphism using the FOG algorithm [21]. The second approach mines frequent subgraphs under the general subgraph isomorphism, using an efficient implementation [2] of the gSpan algorithm [49]. In both cases, the bit-vector encodes the occurrence of the frequent patterns as follows. Given a set of k patterns, each graph g is encoded as a k -dimensional binary vector, where a 1 is marked in the i -th position if the i -th

⁸<http://openbabel.sourceforge.net>

subgraph is subgraph isomorphic to g and a 0 otherwise. We aim at obtaining a similar number of features by selecting an appropriate frequency threshold for the mining algorithms: 4% for FOG, leading to 1376 patterns and 5% for gSpan, leading to 1292 patterns.

We use SVMs as models for these vectors. As implementation we use $\text{SVM}^{\text{light}}$ [23]. For all methods, we use exactly the same settings, which involves applying a polynomial kernel of degree 2, using a 10-fold stratified cross-validation. For each fold, the regularization parameter is tuned by holding out a development set from each training fold of the cross-validation. Finally, we combine the predictions from each test fold, rank all the predictions and compute again the AUROC.

Statistical significance We compute the statistical significance of the different methods in two ways. To estimate the significance of the AUROC comparison between two classifiers, we use the (two-sided) Wilcoxon signed rank test [47], which is a non-parametric alternative to the paired Student’s t-test that does not make any assumption about the distribution of the measurements. In the results, we report the p -value of the test.

To compute statistical comparisons of multiple classifiers over multiple datasets, we use the Friedman test combined with a Nemenyi post-hoc test [11]. The Friedman test is a non-parametric test for statistical comparisons of multiple classifiers. It ranks the algorithms for each dataset separately, with the best performing algorithm getting the rank of 1. In case of a tie, the average rank of the tied models is assigned. Then, a Nemenyi post-hoc test is used to analyse which of the classifier’s ranks differ significantly from each other: the performance is significantly different if the corresponding average ranks differ by at most the critical difference, which depends on the significance level and the number of classifiers [11].

3.3 Results

Q1: What are the differences between MCS-BBP and a state-of-the-art MCS algorithm that uses the general subgraph isomorphism in terms of efficiency and predictive performance? A comparison of the runtimes can be found in Fig. 7. Note that a logarithmic scale is used. The figure clearly shows the exponential behavior of MCS-CAO. For 18 of the 1225 MCS computations, MCS-CAO timed out at 24 hours; we do not show the runtimes of MCS-BBP on these 18 comparisons either. We also report the order statistics of the computation time for both algorithms in Table 1. These show that MCS-BBP computes an $\text{MCS}_{\sqsubseteq}$ in less than 0.713 seconds in 95% of the cases, while MCS-CAO needs 1643 seconds to compute the respective $\text{MCS}_{\sqsubseteq}$. However, in 204 cases, MCS-CAO was faster than MCS-BBP, which indicates that both algorithms are competitive when computing MCSs for smaller molecules. The graph in Fig. 7 confirms this.

While comparing these runtimes, we need to keep in mind that, next to the difference in type of subgraph isomorphism, there is also another difference be-

Figure 7: Runtimes of $\text{MCS}_{\sqsubseteq}$ and $\text{MCS}_{<}$ (in seconds).

tween the MCS algorithms. MCS-CAO computes an unconnected MCS, while the MCS of MCS-BBP is connected. This factor also contributes to the improved runtimes of MCS-BBP and it would be interesting to be able to measure the impact of these two restrictions separately.

Because $\text{MCS}_{<}$ has an exponential behavior, it was not always possible to compute $d_{<}$ while performing k -nearest-neighbour classification. A time-out of 24 hours per computation of one distance (corresponding to one MCS computation) was used. On average, the computation of $d_{<}$ timed out for 0.1% of the molecules. However, since the molecules for which this occurs are large, the denominator in Eq. 1 is large too, which would result in large distances to these molecules. This reduces the chance of them being selected by the k NN algorithm, which indicates that this should not have a big impact on the classification performance.

Figure 8 plots the AUROC of both IBL classifiers. $\text{IBL-}d_{\sqsubseteq}$ scores better than $\text{IBL-}d_{<}$ on all 60 datasets. According to the (two-sided) Wilcoxon signed rank test, this is statistically significant ($p = 1.60 \cdot 10^{-11}$). The average AUROC for $\text{IBL-}d_{\sqsubseteq}$ is 0.760, while for $\text{IBL-}d_{<}$ it is 0.728.

Table 1: Distribution of runtimes of the two MCS algorithms for 1207 comparisons (in seconds).

	$\text{MCS}_{\preceq}$	$\text{MCS}_{\sqsubseteq}$
Average	805.668	0.174
Minimum	0.002	0.002
O.05	0.003	0.003
O.25	0.025	0.012
O.5	0.161	0.049
O.75	3.590	0.238
O.95	1643.850	0.713
Maximum	79625.900	2.951

Conclusion The results of the experimental comparison between MCS-CAO and MCS-BBP show that MCS-BBP outperforms MCS-CAO in terms of efficiency. This can be explained by four restrictions imposed by MCS-BBP: it works with outerplanar graphs, makes use of the BBP subgraph isomorphism, computes connected MCSs and selects a single random one in the case there are multiple MCSs.

Concerning predictive performance, the experimental results indicate that the metric based on the $\text{MCS}_{\sqsubseteq}$ is performing better than the metric based on the $\text{MCS}_{\preceq}$. This means that using the BBP subgraph isomorphism not only leads to a metric that is efficiently computable, but that also has a significantly better predictive performance when used for the classification of molecules. The metric based on the BBP subgraph isomorphism thus seems more meaningful on a chemical level, especially for small pharmacological molecules.

Q2: How does the predictive performance of a metric based on the MCS algorithm using the BBP subgraph isomorphism compare to the predictive performance of state-of-the-art metrics? In Figure 9, we plot the AUROC of $\text{IBL-}d_{\sqsubseteq}$, IBL-FP and IBL-WDK . We find that $\text{IBL-}d_{\sqsubseteq}$ performs consistently better (60 wins out of 60) than the other two methods. According to the Friedman test (Table 2), which shows that the difference in average ranks is larger than the critical difference at 1%, this is statistically significant. Table 2 also shows the average AUROC for the different methods.

Conclusion $\text{IBL-}d_{\sqsubseteq}$ is performing significantly better than the current state-of-the-art metrics for molecules.

Q3: What is the influence on predictive performance of using the BBP subgraph isomorphism as matching operator in other learning techniques? Figure 10 shows a similar comparison between the SVM-based classifiers on the 60 datasets. SVM-FOG scores better than SVM-gSpan on

Figure 8: Predictive performance of the $\text{IBL-}d_{\sqsubseteq}$ and $\text{IBL-}d_{\preceq}$ classifiers on NCI60.

all datasets. According to the (two-sided) Wilcoxon signed rank test, this is statistically significant ($p = 1.60 \cdot 10^{-11}$). The average AUROC for SVM-FOG is 0.762, while for SVM-gSpan it is 0.737.

Conclusion The features generated by the BBP matching operator have a larger predictive power. This can be explained by the fact that SVM-FOG uses a more constrained language, i.e. outerplanar graphs, leading to less redundant patterns. Additional experiments show that it is still possible to boost the performance of SVM-FOG and SVM-gSpan by lowering the support threshold (and in this way obtaining more patterns), but this does not change the above conclusion.

4 Related Work

In this section, we discuss related research from two different backgrounds. First, we focus on algorithms that compute an MCS of two graphs (Sect. 4.1). Second, since we evaluate the metric on a number of classification tasks from chemoinformatics, we also give an overview of general learning methods that have been used for learning quantitative structure-activity relationships in molecules (Sect. 4.2).

Figure 9: Predictive performance of the state-of-the-art IBL classifiers on NCI60.

4.1 MCS algorithms

There exist a number of approaches that compute MCSs of graphs and there are a lot of differences in the way they handle the NP-completeness of the problem. For example, certain algorithms solve the MCS problem optimally, while others do it approximately. Moreover, there exist slightly different notions of an MCS e.g., whether it is connected or whether it is subgraph-induced. We first discuss these different notions.

First, the computed MCS can be connected or unconnected. When computing an unconnected MCS, a trade-off has to be made between the number of connected fragments in the MCS and its usefulness as a pattern. If too many connected components are allowed, the pattern becomes too general. If the number is too small, the pattern may be too specific. For example, when two molecules share two benzene rings but differ with respect to the connection between the rings, only one of these rings can appear in a connected MCS. Despite this potential drawback, we focus on finding connected MCSs here.

A second distinction is made between the maximum common induced subgraph (MCIS) and the maximum common edge subgraph (MCES). The MCIS requires the common subgraph to be induced, in the sense that two vertices in the first graph are linked by an edge if and only if they are mapped to two

Table 2: Average scores and ranks for AUROC for the state-of-the-art IBL classifiers on NCI60.

Method	Average AUROC	Average rank
IBL- $d_{\sqsubseteq}$	0.760	1
IBL-FP	0.742	2.15
IBL-WDK	0.731	2.85
Critical difference for the average ranks at the 1% significance level: 0.53		

vertices in the second graph that are linked by an edge as well.⁹ The MCES, which does not have this requirement, simply tries to maximize the number of edges in the common subgraph. Chemists have argued that the MCES more adequately expresses the notion of chemical similarity than does the MCIS, since the bonded interactions between atoms are most responsible for the perceived molecular activity [38]. An MCS that is computed between outerplanar graphs under the BBP subgraph isomorphism corresponds to neither of these notions, but tries to maximize the edges given that only bridges are mapped to bridges and block edges to block edges.

Here we also introduce a third distinction, which is the kind of subgraph isomorphism used in the matching. Usually, the general subgraph isomorphism is used to compute the MCS, while we propose to use the BBP subgraph isomorphism instead.

An overview of MCS algorithms has been given by Raymond and Willett [38] and Conte et al. [8]. McGregor [32] proposes a simple backtracking strategy that uses various heuristics in order to reduce the number of backtrackings. Akutsu [1] proposes a polynomial algorithm for computing the connected MCES of "almost trees of bounded degree" under the general subgraph isomorphism. This is another class of graphs which is suited for the representation of molecules. Koch [27] introduces an algorithm for enumerating all connected MCESs in two graphs, based on a transformation of the two graphs into an association graph. The MCS problem is then transformed into a clique detection problem. Raymond et al. [36] use the same problem transformation and propose an exact multi-step algorithm which defines a similarity based on computing the unconnected MCES. The problem still remains theoretically NP-complete, but their algorithm makes use of advanced heuristics to reduce the number of matchings required.

Cao et al. [4] propose an improved backtracking algorithm that is specialized for molecules and works directly on the chemical graph. Because of branch-and-bound heuristics, they can drastically reduce the search space and obtain an efficient algorithm for unconnected MCIS computation. They show that their MCS-based similarity measure outperforms traditional descriptor-based

⁹The definition of an induced subgraph G of a graph H (under the mapping φ) requires that for every $u, v \in V(G)$: $\{u, v\} \in E(G)$ iff $\{\varphi(u), \varphi(v)\} \in E(H)$.

Figure 10: Comparison of the performance of the SVM classifiers on NCI60.

and topology-based similarity measures.

Instead of using heuristics, an alternative approach to reduce the complexity could be to consider common substructures which can be computed more easily, such as multisets of common vertex labels [25]. However, not taking into account the more complex shared substructures is an important drawback.

Our algorithm is different from the other approaches in the sense that it computes a connected MCS that fulfils the requirements of the BBP subgraph isomorphism in an exact way. The algorithm works directly on the chemical graph and does not require an advanced graph-theoretical problem transformation.

4.2 QSAR methods

Determining quantitative structure-activity relationships (QSAR) is the process where chemical structure is quantitatively correlated with a biological or chemical activity. Usually, a function is learned that takes as input a number of physicochemical and structural properties of a molecule and then returns its predicted activity. A number of learning approaches have been proposed for QSAR. Depending on the representation they use, we organise them in several categories.

4.2.1 Descriptor-based methods

The first methods for QSAR were *descriptor-based*: using various measurable physicochemical properties (such as polarity or hydrophobicity) of a molecule as descriptors, the molecule can be transformed into a vector of real numbers, each number representing a specific property [16]. Then, a number of propositional similarity measures (e.g., the Tanimoto coefficient [48]) and machine learning methods can be applied.

There are two main difficulties with this approach: (1) domain knowledge is required to select the proper descriptors, and (2) information about the molecular structure is lost. As there is a common agreement that the structure is important when developing similarity measures, there has been an increased interest in relational learning methods.

4.2.2 Relational methods

Because relational methods take structural information directly into account, these methods are often referred to as *topology-based* approaches. One framework in which relational approaches have been developed is that of inductive logic programming (ILP), where the molecules and models for them are described with first-order logic. In this domain, a number of algorithms have been proposed that obtain excellent results on several QSAR tasks [26, 18]. However, because of computational complexity issues, it is still a challenge for ILP algorithms to cope with large datasets.

Actually, the ILP representation is much more general than what is needed to model molecules and to discover substructures. Therefore, a number of methods aim at achieving better performance by building systems that use special-purpose data structures for representing graphs and provide well-chosen primitives for manipulating them (e.g., [49, 35, 7]). In this way, they transform the problem of binding chemical substructures into that of finding subgraphs in a graph. Still, finding subgraphs requires expensive subgraph isomorphism matchings.

A number of such specialised graph algorithms have been developed to compute similarities between graphs [38, 8], such as the MCS algorithms that were discussed in Sect. 4.1. De Raedt and Ramon [10] show that the notion of a maximum common subgraph (or a minimally general generalisation in relational learning terms) can be used to construct a distance measure.

Furthermore, kernels can be viewed as a kind of similarity measure as well. Following the increased emphasis on structure, several kernel functions have been developed that work on graphs directly [20, 15, 44, 5, 43]. Again, the main problem here is to select kernel functions that can capture the molecule’s activity while still remaining efficient to compute. Many types of graph kernels correspond to some implicit feature space where features are generated for certain subgraphs or graph properties. The kernel then counts how many subgraphs two examples have in common. The subgraphs are typically defined in such a way that they can be enumerated implicitly and such that the counting proce-

dures can be done efficiently (e.g., via dynamic programming procedures). For example, Ceroni et al. [5] have introduced the weighted decomposition kernel (WDK), which obtains state-of-the-art results when used for the classification of molecules. It is based on a decomposition of the molecule in a selector (a single vertex) and a context (a fixed-radius subgraph surrounding the selector). By selecting an appropriate kernel for these structures, the computation of the WDK kernel remains feasible.

4.2.3 Propositionalization methods

To avoid having to deal with subgraph isomorphisms during the learning phase, a popular approach in relational learning is to propositionalize the graph-based representation into an attribute-valued representation [9]. In this context, the learning process involves two separate steps: first, the complete dataset of graphs is searched for subgraphs and then, each example is represented as a bit-vector, which is often called *fingerprint*, that encodes the occurrences of these subgraphs in the example (see Sect. 3.2). The difference with the above mentioned descriptor-based methods is that the vectors here encode topological features instead of numerical ones, trying to minimize the loss of structural information. Since the graph miner automatically selects the "most interesting" features, this approach solves the first difficulty mentioned above, while the second difficulty is transformed into selecting a suitable criterion that finds the most interesting features, i.e., the patterns that describe the molecule's topology best in order to discriminate molecules w.r.t. their activity.

To this end, various techniques have been used to generate features of interest [29]. In the chemoinformatics context, there are two important approaches: either the patterns are generated exhaustively, or some kind of constraint is used to select the most interesting patterns.

The current state-of-the-art methods towards feature construction for molecules generate all sequences [48] or subgraphs [46] of size up to k that occur in at least one molecule. Because even for small values of k , this rapidly leads to vast numbers of features, the generated features are typically compressed using some kind of hashing of the occurrences of the paths onto a fixed-length vector. Because of its size, the resulting pattern set is hard to interpret, and the use of hashing methods only aggravates this problem.

In order to limit the size of the pattern set, a number of constraint-based graph mining methods have been proposed, where constraints on the subgraphs of interest are formulated. A wide variety of different constraints has been considered, such as frequency [12, 46], statistical significance measures such as χ^2 [2, 17], syntactical constraints [28], randomization [6] or a combination of those [31]. As mentioned in the introduction, Horváth et al. [21] proposed to use the BBP subgraph isomorphism on outerplanar graphs, yielding a set of patterns that are topographically restricted and can be mined more efficiently. In these types of approaches, one typically performs a complete search. This leads one to finding all patterns satisfying the constraints.

5 Conclusions and Future Work

In this article, we have introduced an algorithm computing the MCS of two outerplanar graphs under the BBP subgraph isomorphism. We have proved its correctness and shown that it runs in polynomial time, both theoretically and empirically.

The polynomial time complexity is realized by four restrictions. First, the algorithm only works on outerplanar graphs. Most molecular graphs, which are of most interest to us, are outerplanar. Second, the algorithm uses the BBP subgraph isomorphism as matching operator. Again, in the chemical context, this seems a sensible choice. Third, we only consider connected MCSs. Fourth, in the case there are multiple possible MCSs, we only select one randomly.

We have shown that our algorithm outperforms a state-of-the-art MCS algorithm that computes an unconnected MCS under the general subgraph isomorphism, both in terms of efficiency and predictive performance. To evaluate the predictive performance, we have constructed a metric based on the MCS and used it in an instance-based learning classification task.

We have also investigated our MCS-based metric and it turns out that this BBP-based metric outperforms previously published metrics. One reason may be that dealing differently with cycles and linear fragments makes sense in chemical applications. Moreover, it is more intuitive than other metrics and graph kernels and it uses the original graph structure.

Finally, we have investigated the performance of the BBP matching operator in general. Experimental results show that the BBP matching operator obtains good performance as well when used to generate features.

We can conclude that, at least for molecular datasets, the BBP matching operator can, next to the gain in efficiency, also improve the predictive performance of graph mining techniques and hence, it is an interesting matching operator for molecules. Therefore, depending on the situation (e.g., the number of examples) and the user’s preferences (e.g., the interpretability of the predictions), either a BBP-metric (IBL) or BBP-generated features can be good choices.

Since a fraction of the molecules in the NCI database cannot be represented by outerplanar graphs, it is useful to investigate how the ideas of the BBP matching operator can be extended to non-outerplanar graphs. It would also be interesting to run our algorithm on other outerplanar datasets such as RNA molecules and to investigate other classes of graphs which would be suitable to represent molecules (e.g., graphs having a bounded treewidth) and for which we can design polynomial algorithms. Finally, comparing with similar algorithms, such as the one of Akutsu [1], remains interesting future work.

Acknowledgements

Leander Schietgat is supported by the ERC Starting Grant 240186 “MiGraNT” and the Research Fund KU Leuven. Jan Ramon is supported by the ERC Starting Grant 240186 “MiGraNT” and the KU Leuven GOA project “Probabilistic

Logic Learning”. This research was also supported by FWO Vlaanderen. We thank Fabrizio Costa for valuable discussions and Mostafa Haghir Chehreghani for proofreading.

References

- [1] T. Akutsu. A polynomial time algorithm for finding a largest common subgraph of almost trees of bounded degree. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science*, E76-A:1488–1493, 1993.
- [2] B. Bringmann, A. Zimmermann, L. De Raedt, and S. Nijssen. Don’t be afraid of simpler patterns. In *Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 55–66, 2006.
- [3] H. Bunke and K. Shearer. A graph distance metric based on the maximal common subgraph. *Pattern Recognition Letters*, 19(3-4):255–259, 1998.
- [4] Y. Cao, T. Jiang, and T. Girke. A maximum common substructure-based algorithm for searching and predicting drug-like compounds. *Bioinformatics*, 24(13):i366–i374, 2008.
- [5] Alessio Ceroni, Fabrizio Costa, and Paolo Frasconi. Classification of small molecules by two- and three-dimensional decomposition kernels. *Bioinformatics*, 23(16):2038–2045, 2007.
- [6] Vineet Chaoji, Mohammad Al Hasan, Saeed Salem, Jeremy Besson, and Mohammed J. Zaki. Origami: A novel and effective approach for mining representative orthogonal graph patterns. *Stat. Anal. Data Min.*, 1(2):67–84, 2008.
- [7] Yun Chi, Richard R. Muntz, Siegfried Nijssen, and Joost N. Kok. Frequent subtree mining - an overview. *Fundam. Inform.*, 66(1-2):161–198, 2005.
- [8] D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(3):265–298, 2004.
- [9] L. De Raedt. *Logical and Relational Learning*. Springer, 2008.
- [10] L. De Raedt and J. Ramon. Deriving distance metrics from generality relations. *Pattern Recognition Letters*, 30(3):187–191, 2009.
- [11] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7(Jan):1–30, 2006.
- [12] M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis. Frequent substructure-based approaches for classifying chemical compounds. *IEEE Transactions on Knowledge and Data Engineering*, 17(8):1036–1050, 2005.

- [13] Reinhard Diestel. *Graph Theory*. Springer-Verlag, 2000.
- [14] Michael R. Garey and D.S. Johnson. *Computers and Intractability: a Guide to the theory of NP-Completeness*. Freeman and Co., 1979.
- [15] Thomas Gärtner. *Kernels for Structured Data*. World Scientific, 2008.
- [16] C. Hansch, P.P. Maolney, T. Fujita, and Muir R.M. Correlation of biological activity of phenoxyacetic acids with hammett substituent constants and partition coefficients. *Nature*, 194:178–180, 1962.
- [17] Huahai He and Ambuj K. Singh. Graphrank: Statistical modeling and mining of significant subgraphs in the feature space. In *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, pages 885–890, Washington, DC, USA, 2006. IEEE Computer Society.
- [18] C. Helma, Kramer S., and De Raedt L. Data mining and machine learning techniques for the identification of mutagenicity inducing substructures and structure activity relationships of noncongeneric compounds. *Journal of Chemical Information and Computer Systems*, 44(4):1402–141, 2004.
- [19] J.E. Hopcroft and R.M. Karp. A $n^{5/2}$ algorithm for maximum matching in bipartite graphs. *SIAM Journal of Computing*, 2:225–231, 1973.
- [20] Tamás Horváth, Thomas Gärtner, and Stefan Wrobel. Cyclic pattern kernels for predictive graph mining. In *KDD '04: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge discovery and Data Mining*, pages 158–167, 2004.
- [21] Tamás Horváth, Jan Ramon, and Stefan Wrobel. Frequent subgraph mining in outerplanar graphs. In *KDD '06: Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 197–206, Philadelphia, PA, August 2006.
- [22] Tamás Horváth, Jan Ramon, and Stefan Wrobel. Frequent subgraph mining in outerplanar graphs. *Knowledge Discovery and Data Mining*, 21(3):472–508, 2010.
- [23] T. Joachims. *Learning to Classify Text using Support Vector Machines: Methods, Theory, and Algorithms*. Springer, 2002.
- [24] M.A. Johnson and G.M. Maggiora. *Concepts and Applications of Molecular Similarity*. John Wiley, 1990.
- [25] T. Karunaratne and H. Boström. Learning to classify structured data by graph propositionalization. In *Proceedings of the Second IASTED International Conference on Computational Intelligence*, pages 393–398, 2006.

- [26] R.D. King, S. Muggleton, A. Srinivasan, and M.J.E. Sternberg. Structure-activity relationships derived by machine learning: The use of atoms and their bond connectivities to predict mutagenicity by inductive logic programming. *Proceedings of the National Academy of Sciences*, 93:438–442, 1996.
- [27] Ina Koch. Enumerating all connected maximal common subgraphs in two graphs. *Theor. Comput. Sci.*, 250(1-2):1–30, 2001.
- [28] S. Kramer, L. De Raedt, and C. Helma. Molecular feature mining in HIV data. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-01)*, pages 136–143. ACM Press, 2001.
- [29] S. Kramer, N. Lavrač, and P. Flach. Propositionalization approaches to relational data mining. In S. Džeroski and N. Lavrač, editors, *Relational Data Mining*, pages 262–291. Springer-Verlag, 2001.
- [30] A. Lingas. Subgraph isomorphism for biconnected outerplanar graphs in cubic time. *Theoretical Computer Science*, 63:295–302, 1989.
- [31] Andreas Maunz, Christoph Helma, and Stefan Kramer. Large-scale graph mining using backbone refinement classes. In *KDD '09: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 617–626, New York, NY, USA, 2009. ACM.
- [32] J. J. McGregor. Backtrack search algorithms and the maximal common subgraph problem. *Software: Practice and Experience*, 12:23–34, 1982.
- [33] Sandra L. Mitchell. Linear algorithms to recognize outerplanar and maximal outerplanar graphs. *Information Processing Letters*, 9(5):229–232, 1979.
- [34] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38, 1957.
- [35] Siegfried Nijssen and Joost N. Kok. A quickstart in frequent structure mining can make a difference. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 647–652, 2004.
- [36] J. Raymond, E. Gardiner, and P. Willett. Rascal: Calculation of graph similarity using maximum common edge subgraphs. *The Computer Journal*, 45:631–644, 2002.
- [37] J. Raymond and P. Willett. Effectiveness of graph-based and fingerprint-based similarity measures for virtual screening of 2D chemical structure databases. *Journal of Computer-Aided Design*, 16:59–71, 2002.

- [38] J. Raymond and P. Willett. Maximum common subgraph isomorphism algorithms for the matching of chemical structures. *Journal of Computer-Aided Molecular Design*, 16:521–533, 2002.
- [39] L. Schietgat, J. Ramon, M. Bruynooghe, and H. Blockeel. An efficiently computable graph-based metric for the classification of small molecules. In *Proceedings of the 11th International Conference on Discovery Science*, volume 5255 of *Lecture Notes in Artificial Intelligence*, pages 197–209, 2008.
- [40] Leander Schietgat, Fabrizio Costa, Jan Ramon, and Luc De Raedt. Effective feature construction by maximum common subgraph sampling. *Machine Learning*, 83(2):137–161, 2011.
- [41] Ron Shamir and Dekel Tsur. Faster subtree isomorphism. *Journal of Algorithms*, 33(2):267–280, November 1992.
- [42] K. Shearer, H. Bunke, and S. Venkatesh. Video indexing and similarity retrieval by largest common subgraph detection using decision trees. *Pattern Recognition*, 34(5):1075–1091, 2001.
- [43] Nino Shervashidze and Karsten Borgwardt. Fast subtree kernels on graphs. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1660–1668, 2009.
- [44] S. Joshua Swamidass, Jonathan Chen, Jocelyne Bruand, Peter Phung, Liva Ralaivola, and Pierre Baldi. Kernels for small molecules and the prediction of mutagenicity, toxicity and anti-cancer activity. *Bioinformatics*, 21(suppl.1):i359–368, 2005.
- [45] M. Syslo. The subgraph isomorphism problem for outerplanar graphs. *Theoretical Computer Science*, 17(1):91–97, 1982.
- [46] N. Wale, I.A. Watson, and G. Karypis. Comparison of descriptor spaces for chemical compound retrieval and classification. *Knowledge and Information Systems*, 14:347–375, 2008.
- [47] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics*, 1:80–83, 1945.
- [48] P. Willett. Similarity-based virtual screening using 2D fingerprints. *Drug Discovery Today*, 11(23/24):1046–1051, 2006.
- [49] Xifeng Yan and Jiawei Han. gSpan: Graph-based substructure pattern mining. In *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002)*, pages 721–724, Japan, 2002. IEEE Computer Society.