Online Structural Graph Clustering Using Frequent Subgraph Mining

Madeleine Seeland, Tobias Girschick, Fabian Buchwald, and Stefan Kramer

Technische Universität München,
Institut für Informatik/I12,
Boltzmannstr. 3, 85748 Garching b. München, Germany
{madeleine.seeland,tobias.girschick,
fabian.buchwald,stefan.kramer}@in.tum.de

Abstract. The goal of graph clustering is to partition objects in a graph database into different clusters based on various criteria such as vertex connectivity, neighborhood similarity or the size of the maximum common subgraph. This can serve to structure the graph space and to improve the understanding of the data. In this paper, we present a novel method for structural graph clustering, i.e. graph clustering without generating features or decomposing graphs into parts. In contrast to many related approaches, the method does not rely on computationally expensive maximum common subgraph (MCS) operations or variants thereof, but on frequent subgraph mining. More specifically, our problem formulation takes advantage of the frequent subgraph miner gSpan (that performs well on many practical problems) without effectively generating thousands of subgraphs in the process. In the proposed clustering approach, clusters encompass all graphs that share a sufficiently large common subgraph. The size of the common subgraph of a graph in a cluster has to take at least a user-specified fraction of its overall size. The new algorithm works in an online mode (processing one structure after the other) and produces overlapping (non-disjoint) and non-exhaustive clusters. In a series of experiments, we evaluated the effectiveness and efficiency of the structural clustering algorithm on various real world data sets of molecular graphs.

1 Introduction

Mining graph data has attracted a lot of attention in the past ten years [1–3]. One family of methods is concerned with mining subgraph patterns in graph databases [1, 2]. The criteria for interestingness are often based on the support of a pattern in the graph database, e.g., requiring a minimum and / or maximum frequency, closedness, freeness or class-correlation. However, in all of these cases, the structural diversity of graph databases, i.e. the existence of groups of similar or dissimilar graphs, is not explicitly taken into account or revealed by the algorithm. Vice versa, the structural composition and existence of groups of similar graphs has a serious impact on the output and runtime performance of

J.L. Balcázar et al. (Eds.): ECML PKDD 2010, Part III, LNAI 6323, pp. 213–228, 2010.

pattern mining algorithms. To gain insights into the structural characteristics of graph data sets, we developed a graph clustering algorithm that discovers groups of structurally similar and dissimilar graphs. The algorithm can be practically useful for a variety of purposes: benchmarking other graph mining algorithms, descriptor calculation (e.g., for QSAR studies), computing local models for classification or regression (e.g., one per cluster) and calculation of the so-called applicability domain of models.

To illustrate the impact of structural diversity on graph mining results, we consider two data sets of molecular graphs of the same size and with approximately the same number of atoms per molecule. The first data set, the COX2 data set (`http://pubs.acs.org/doi/suppl/10.1021/ci034143r`), contains 414 compounds, which possess a relatively high structural homogeneity. The second data set is a subset of the CPDB data set (`http://potency.berkeley.edu/`) that matches the COX2 data both in the number of structures and the number of atoms per structure. The results of a typical graph mining representative, gSpan, and the results of the graph clustering algorithm presented in this paper are shown in Figure 1. In the upper part of the figure, we see the huge difference in the runtime and the number of discovered patterns. For structurally homogeneous data (COX2), the number of patterns and runtime explodes, whereas for structurally heterogeneous data (CPDB) the algorithm behaves as expected. The reason for this difference in performance becomes evident in the graph clustering results in the lower part of Figure 1. As can be seen, there is a small number of large clusters in COX2 and a large number of small clusters in CPDB (for each value of a parameter that is varied on the x-axis). This indicates a high degree of structural homogeneity in COX2 and a low degree in CPDB, and also hints at the usefulness of graph clustering to make the characteristics of a graph database explicit.

The graph clustering algorithm presented in this paper operates directly on the graphs, i.e. it does not require the computation of features or the decomposition into subgraphs. It works online (processing one graph after the other) and creates a non-disjoint and non-exhaustive clustering: graphs are allowed to belong to several clusters or no cluster at all. One important component of the algorithm is a variant of gSpan to determine cluster membership. Thus, the proposed graph clustering approach is based on a practically fast graph mining algorithm and not on typically time-consuming maximum common subgraph (MCS) operations [4]. In contrast to another graph clustering approach based on graph pattern mining [5], the (often quite numerous) frequent subgraphs are just by-products, and not part of the output of the algorithm: the actual output consists just of the clustered graphs sharing a common scaffold.

The remainder of the paper is organized as follows. In Section 2 the methodology of our structure-based clustering algorithm is introduced. Section 3 presents a description of the data sets and experiments as well as an interpretation of the results. In Section 4 related work is discussed before Section 5 gives a conclusion and an outlook to future work.

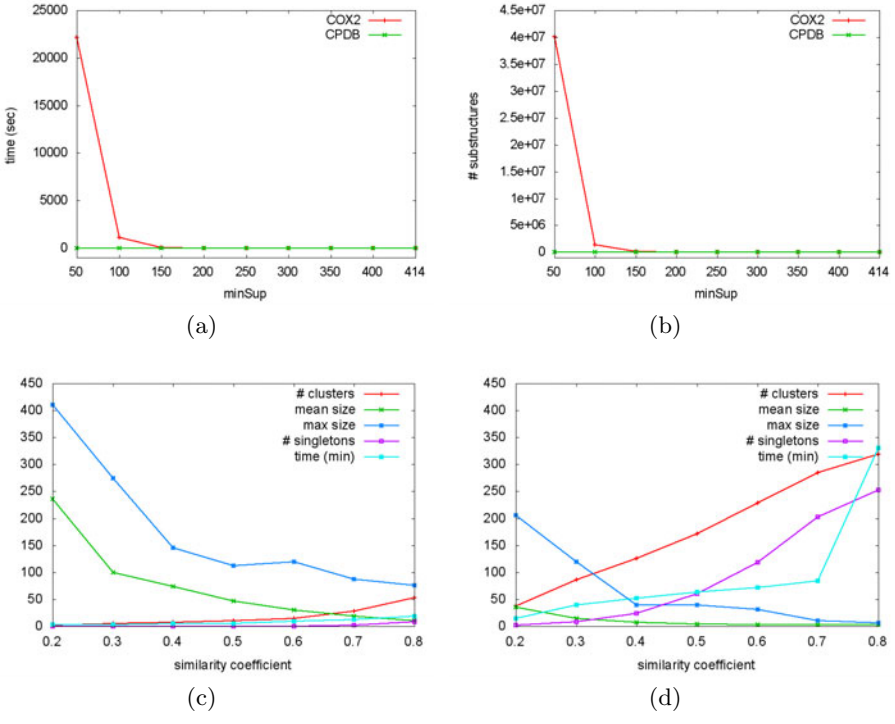


Fig. 1. (Above) (a) Runtime behavior and (b) number of subgraphs for gSpan on COX2 and CPDB. (Below) Results of structural clustering on (c) COX2 and (d) CPDB.

2 Structural Clustering

The following section presents the structural clustering algorithm that can be used to cluster graph instances based on structural similarity. Starting with some definitions from graph theory, we present the problem definition and the algorithm in detail.

2.1 Notation and Definitions

In the following, all graphs are assumed to be labeled, undirected graphs. To be more precise, a graph and its subgraphs are defined as follows: A labeled *graph* is represented as a 4-tuple $g = (V, E, \alpha, \beta)$, where V is a set of vertices and $E \subseteq V \times V$ is a set of edges representing connections between all or some of the vertices in V. $\alpha : V \rightarrow L$ is a mapping that assigns labels to the vertices, and $\beta : V \times V \rightarrow L$ is a mapping that assigns labels to the edges. Given two labeled graphs $g = (V, E, \alpha, \beta)$ and $g' = (V', E', \alpha', \beta')$, g' is a *subgraph* of g, $(g' \subseteq g)$ if:

- $V' \subseteq V$
- $E' \subseteq E$

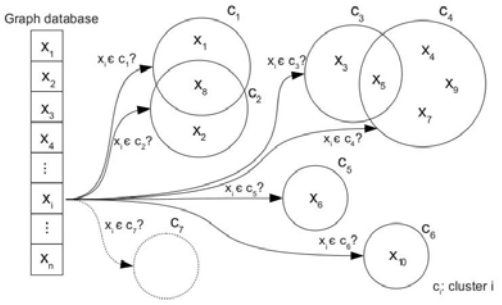


Fig. 2. Schematic overview of the cluster membership assignment for instance x_i. Graph instances are represented by $x_1, ..., x_n$, clusters by $C_1, ..., C_k$.

- $\forall x \in V' : \alpha'(x) = \alpha(x)$
- $\forall (x, y) \in V' \times V' : \beta'((x, y)) = \beta((x, y))$

Given two arbitrary labeled graphs $g_1 = (V_1, E_1, \alpha_1, \beta_1)$ and $g_2 = (V_2, E_2, \alpha_2, \beta_2)$, a *common subgraph* of g_1 and g_2, $cs(g_1, g_2)$, is a graph $g = (V, E, \alpha, \beta)$ such that there exists a *subgraph isomorphism* from g to g_1 and from g to g_2. This can be generalized to sets of graphs. The set of common subgraphs of a set of graphs $\{g_1, ..., g_n\}$ is then denoted by $cs(\{g_1, ..., g_n\})$. Moreover, given two graphs g_1 and g_2, a graph g is called a *maximum common subgraph* of g_1 and g_2 if g is a common subgraph of g_1 and g_2 and there exists no other common subgraph of g_1 and g_2 that has more vertices than g. Finally, we define the size of a graph as the number of its vertices, i.e. $|V|$.

2.2 Problem Definition

Structural clustering is the problem of finding groups of graphs sharing some structural similarity. Instances with similar graph structures are expected to be in the same cluster provided that the common subgraphs match to a satisfactory extent. Only connected subgraphs are considered as common subgraphs. The similarity between graphs is defined with respect to some user-defined size threshold. The threshold is set such that the common subgraphs shared among a query graph and all cluster instances make up a specific proportion of the size of each graph. A graph is assigned to a cluster provided that there exists at least one such common subgraph whose size is equal or bigger than the threshold. In this way, an object can simultaneously belong to multiple clusters (overlapping clustering) if the size of at least one common subgraph with these clusters is equal or bigger than the threshold. If an object does not share a common subgraph with any cluster that meets the threshold, this object is not included in any cluster (non-exhaustive clustering). A graphical overview is shown in Figure 2. For one graph after the other, it is decided whether it belongs to an existing cluster or whether a new cluster is created.

Formally, we frame the problem of structural clustering as follows. Given a set of graph objects $X = \{x_1, ..., x_n\}$, we need to assign them into clusters which may overlap with each other. In clustering these objects, one objective is considered: to maximize the average number of objects contained in a cluster, such that at any time for each cluster C there exists at least one common subgraph that makes up a specific proportion, θ, of the size of each cluster member. Considering the state of a cluster $C = \{x_1, ..., x_m\}$[1] at any point in time, the criterion can formally be defined as:

$$\exists\, s \in cs(\{x_1, ..., x_m\}) \forall x_i \in C : |s| \geq \theta |x_i| \tag{1}$$

where s is a subgraph and $\theta \in [0, 1]$ is a user-defined similarity coefficient. According to this goal, a minimum threshold for the size of the common subgraphs shared by the query graph x_{m+1} and the instances in cluster C can be defined as

$$minSize = \theta\ max(|x_{max}|, |x_{m+1}|), \tag{2}$$

where $\theta \in [0, 1]$ and x_{max} is the largest graph instance in the cluster. To obtain meaningful and interpretable results, the minimum size of a graph considered for cluster membership is further constrained by a *minGraphSize* threshold. Only graphs whose size is greater than *minGraphSize* are considered for clustering. Thus, the identification of the general cluster scaffold will not be impeded by the presence of a few graph structures whose scaffold is much smaller than the one the majority of the cluster members share. This will be especially useful in real-world applications that often contain small fragments (see the minimum size column in Table 1).

2.3 Algorithm

The clustering algorithm works as follows. Let *minGraphSize* be the minimum threshold for the graph size and *minSize* be the minimum threshold for the size of the common subgraphs specified by the user and defined in Equation 2. In the first step, an initial cluster is created containing the first graph object that is larger than *minGraphSize*. In the following steps, each instance is compared against all existing clusters. In case the query instance meets the *minGraphSize* threshold and shares at least one common subgraph with one or more clusters that meets the cluster criterion in Equation 2, the instance is added to the respective cluster. Unlike many traditional clustering algorithms, an object is allowed to belong to no cluster, since it is possible that an object is not similar to any cluster. Thus, in this case, a new singleton cluster is created containing the query instance. The proposed clustering algorithm has two main advantages over many clustering algorithms. First, the algorithm works in an online mode, since it does not keep all the examples in memory at the same time, but processes them one by one in a single pass. Second, in contrast to many clustering algorithms which assume that the number of clusters is known beforehand, our algorithm does not require the specification of the number of clusters a priori.

[1] In slight abuse of notation, we use the same indices as above.

Algorithm 1. Structural Clustering

```
1: // graph[] - array of n graphs to be clustered
2: // θ - similarity coefficient (θ ∈ [0, 1])
3: // minGraphSize - minimum graph size
4: procedure SC(graph[], θ, minGraphSize)
5:     // Initialize the clustering with graph[0]
6:     clusters[] ← ∞
7:     // loop over all graphs
8:     for (j ← 0, n) do
9:         hasCluster ← false
10:        if (graph[j] ≥ minGraphSize) then
11:            // compare graph against all existing clusters
12:            for all c ∈ clusters[] do
13:                minSize ← θ · max(size(graph[j]), size(c.max))
14:                // check for cluster exclusion criteria defined in Equation 3 and 4
15:                if (3) || (4) then
16:                    continue
17:                else
18:                    minSup ← c.size + 1
19:                    // add graph[j] to cluster c if gSpan finds at least one
20:                    // common subgraph that meets the minSize threshold
21:                    if gSpan''(graph[j] ∪ c.graphs, minSup, minSize) then
22:                        c[last + 1] ← graph[j]
23:                        hasCluster ← true
24:                    end if
25:                end if
26:            end for
27:            // create new cluster if the graph was not clustered
28:            if (hasCluster = false) then
29:                clusters[last + 1] ← newCluster(graph[j])
30:            end if
31:        end if
32:    end for
33: end procedure
```

The pseudocode for the structural clustering algorithm is shown in Algorithm 1.

For computing common subgraphs, we use a modified version of the graph mining algorithm gSpan [2] that mines frequent subgraphs in a database of graphs satisfying a given minimum frequency constraint. In this paper, we require a minimum support threshold of $minSup = 100\%$ in a set of graphs, i.e. all common subgraphs have to be embedded in all cluster members. For our experiments with molecular graph data, we use gSpan', an optimization of the gSpan algorithm for mining molecular databases (`http://wwwkramer.in.tum.de/projects/gSpan.tgz`). Since we do not need to know all common subgraphs of a set of graphs, but rather only want to find out if there exists at least one common subgraph that meets the minimum size threshold defined in Equation 2, it is possible to terminate search once a solution

is found. Due to the structural asymmetry of the search tree (all descendants of a subgraph are generated before its right siblings are extended), it is thus possible to modify gSpan' such that the procedure exits immediately when a common subgraph is found that satisfies the minimum size threshold defined in Equation 2. In this way, a substantial improvement in runtime performance can be achieved. In the pseudocode, this modification of gSpan' is called gSpan". We introduced a special label for edges in cyclic graphs to ensure that cylic graph structures are not subdivided any further. Moreover, we introduced the following two cluster exclusion criteria to avoid unnecessary calls to the gSpan" algorithm:

$$|x_{m+1}| > |x_{max}| \wedge minSize > |x_{min}| \tag{3}$$

$$|x_{m+1}| < |x_{min}| \wedge minSize > |x_{m+1}|, \tag{4}$$

where x_{min} is the smallest graph in cluster C and x_{m+1} and x_{max} are defined as above. Due to these exclusion criteria, graph instances which cannot fulfill the minimum subgraph size threshold are eliminated from further consideration. The first criterion (3) excludes too large query instances that would break up an existing cluster while the second one (4) excludes too small query instances. In case at least one of the two exclusion citeria is met, we omit the computation of the common subgraphs and continue with the next cluster comparison.

In summary, three factors contribute to the practically favorable performance of the approach: First, the use of a gSpan variant to compute a sufficiently large common subgraph, which is known to be effective on graphs of low density. Second, the possibility to terminate search as soon as such a subgraph is found. Third, the cluster exclusion criteria to avoid unnecessary runs of gSpan".

3 Experiments

To evaluate the the effectiveness and efficiency of the new structure-based clustering approach introduced in Section 2.3, we conducted several experiments on eight publicly available data sets of molecular graphs (Table 1). In this section, we describe the experimental set-up and the results.

Baseline Comparison with Fingerprint Clustering The structure-based clustering algorithm was compared with a baseline clustering algorithm based on fingerprint similarity. The goal of this experiment is to determine if our algorithm is able to increase cluster homogeneity as compared to fingerprint clustering. Fingerprint-based similarities can be calculated extremely fast and have been found to perform reasonably well in practice. For the fingerprint calculation of the molecular graph data, the chemical fingerprints in Chemaxon's JChem Java package are used. The Tanimoto coefficient is used as similarity measure between fingerprints, since these fingerprints are equivalent to Daylight fingerprints (`http://www.daylight.com/dayhtml/doc/theory/theory.finger.html`) which were shown to work well in combination with the Tanimoto coefficient [4, 6].

Table 1. Overview of the data sets used for assessing the structural clustering method

Short-hand	n	min size	mean size	max size
$CPDBAS\_MOUSE^a$	444	2	13	64
$CPDBAS\_RAT^a$	580	2	14	90
CYP^b	700	1	24	86
$NCIanti-HIV^c$	36255	3	25	139
$SACA^d$	107	5	27	79
$EPAFHM^e$	580	2	10	55
$FDAMDD^f$	1216	3	23	90
$RepDose^g$	590	2	10	88

[a] http://epa.gov/NCCT/dsstox/sdf_cpdbas.html [b] http://pubs.acs.org/doi/suppl/10.1021/ci0500536
[c] http://dtp.nci.nih.gov/docs/aids/aids_data.html [d] http://dtp.nci.nih.gov/docs/cancer/searches/standard_mechanism.html [e] http://epa.gov/NCCT/dsstox/sdf_epafhm.html
[f] http://epa.gov/NCCT/dsstox/sdf_fdamdd.html [g] http://www.fraunhofer-repdose.de/

The fingerprint-based clustering (FP clustering) works as follows. Iteratively, each molecular graph is compared against all yet existing clusters. In case the query graph meets a predefined minimum graph size threshold, $minGraphSize$, and exceeds the minimum accepted Tanimoto similarity coefficient compared to each graph in the cluster, the query graph is added to the respective cluster; otherwise a new singleton cluster is created containing the query graph. As the FP does not provide a measure for the size of the shared subgraph between the FP cluster members, the common cluster scaffold is obtained by calculating the MCS common to all members in order to get a comparison metric for the proportion of the common subgraph. Note that this step can be omitted in our structural clustering approach, since the defined similarity coefficient provides a measure for the proportion of the common subgraph. In case a graph i is added to a cluster, the MCS between i and the current MCS of the cluster j is calculated. This MCS is iteratively reduced in size as it is compared to the new cluster members that may not share the entire subgraph. For the MCS calculation the maximum common edge subgraph algorithm was used which is implemented in Chemaxon's JChem java package.

Our structural clustering approach was compared with the baseline FP clustering method on the data sets in Table 1. Due to space limitations, we present the results on three representative data sets, CPDBAS_MOUSE, CPDBAS_RAT and EPAFHM. In all experiments, we performed structural clustering for $\theta \in [0.2, 0.8]$. For FP clustering we used a Tanimoto coefficient value in the range of $[0.4, 0.8]$. Due to the different input parameters of both clustering approaches, it is not obvious how to compare the clustering results. However, the clustering statistics in Figure 3 suggest a correlation between the results from structural clustering for a similarity coefficient value of x ($x \in [0, 1]$) and the results from FP clustering for a Tanimoto similarity value of $y = x - 0.1$ ($y \in [0, 1]$), due to similar clustering results in terms of the number of clusters, the number of singletons and the mean and maximum size of the clusters. Thus, we compare the clustering results from both algorithms with respect to this heuristic. Figure

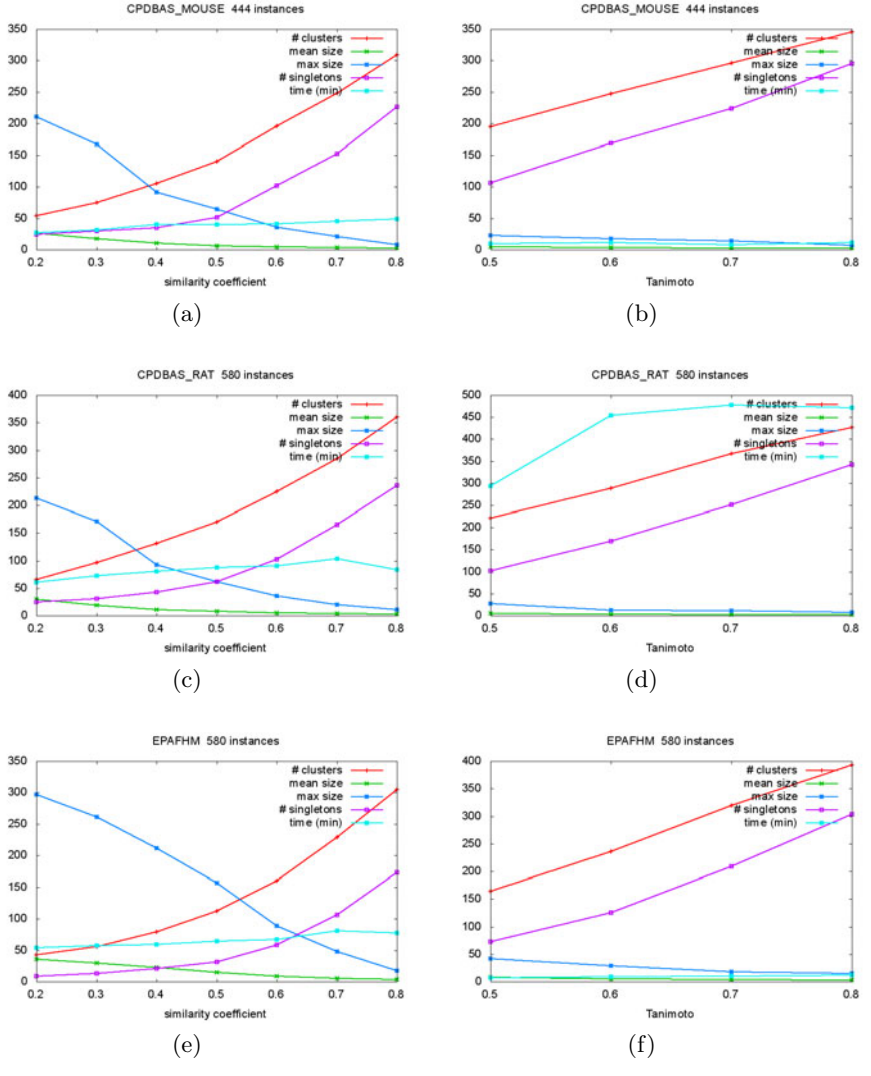


Fig. 3. Results of structural clustering (a), (c), (e) vs. fingerprint clustering (b), (d), (f) on CPDBAS_MOUSE, CPDBAS_RAT and EPAFHM

4 shows the histogram of the share of the MCS of the largest cluster instance for all non-singleton FP clusters for a sample Tanimoto coefficient value of 0.6. The results indicate that the MCS size proportions are, in many cases, below the according structural similarity coefficient θ, which serves as a lower bound on the MCS size proportion. In contrast, each cluster obtained by structural clustering contains at least one common subgraph whose share of each cluster member is equal or bigger than θ. The results suggest that, in comparison to FP clustering,

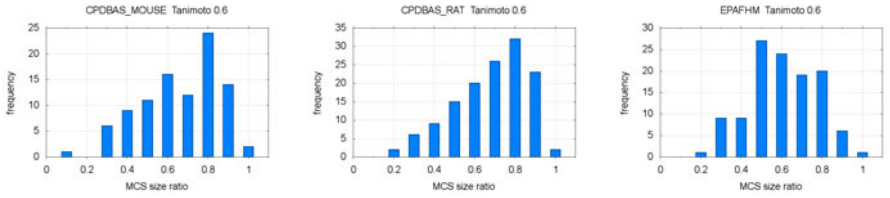


Fig. 4. Histogram of the share of the MCS of the largest cluster instance for fingerprint clustering on (a) CPDBAS_MOUSE, (b) CPDBAS_RAT and (c) EPAFHM using a Tanimoto coefficient value of 0.6.

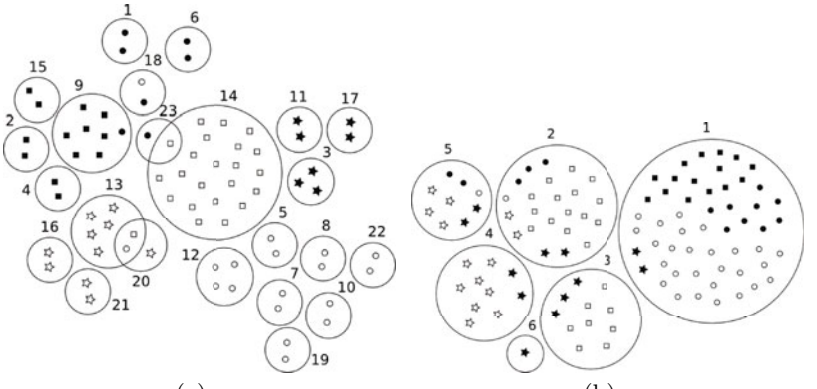
structural clustering provides a superior clustering with reduced heterogeneity in the individual clusters in the overall clustering.

Qualitative Analysis of Structure-Based Clustering. Cluster analysis was performed on the data set of 107 standard anti-cancer agents (SACA) whose class labels corresponding to their mechanisms of action have been clearly classified [7, 8]. The purpose of the experiment was to test if the clusters obtained by structural clustering were in good agreement with the known SACA class labels. As an external measure for clustering validation we used the Rand index to quantify the agreement of the clustering results with the SACA classes. Larger values for the Rand index correspond to better agreement between the clustering results and the SACA classes, with 1.0 indicating perfect concordance. Table 2 shows the Rand index values for different similarity coefficient values. Structural clustering clearly shows the peak point of the Rand index at $\theta = 0.6$. In the following, we present the clustering results for $\theta = 0.6$ partitioning the 107 agents into 52 clusters. 23 of these clusters have at least two members, while the final 29 clusters consist of a single graph. Figure 5(a) gives a representation of the structural clusters with at least two instances in a hypothetical (non-Euclidean) two-dimensional (descriptor) space, where large circles represent clusters and dots, rectangles and stars denote cluster members according to the SACA classes. The results indicate that the clusters tend to be associated with certain SACA classes. Across different values for θ we observed that with a higher similarity coefficient a finer but cleaner grouping of the structures at the cost of generating a larger number of smaller clusters is achieved. The graphs in each class are more cleanly discriminated from other graphs in the data set. Moreover, the clustering produces less overlapping clusters with internally higher structural similarity. In summary, structural clustering is capable of effectively grouping the 107 agents. Graphs instances from the same cluster not only share common subgraphs but are also strongly associated with specific SACA classes of mechanisms of action.

Graph Clustering Comparison Method. Our method was compared with a graph-based clustering based on variational Dirichlet process (DP) mixture models and frequent subgraph mining by Tsuda and Kurihara [5]. This clustering approach addresses the problem of learning a DP mixture model in the

Table 2. Number of clusters and Rand index values for structural clustering on SACA

θ	0.02	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
# Clusters	6	7	11	22	32	39	48	52	60	66	82
Rand Index	0.408	0.436	0.515	0.765	0.827	0.854	0.863	0.869	0.866	0.848	0.833



(a) (b)

Fig. 5. Results of (a) structural clustering for $\theta = 0.6$ and (b) DP Clustering for $\alpha = 0.1$ and $m = 1000$ on the SACA data set. The different symbols for the cluster instances represent the six SACA classes.

high dimensional feature space of graph data. We investigated if the approach is also able to rediscover the known structure classes in the SACA database. In this experiment, we varied the number of features m from 50 to 5000 and set $\alpha = 0.01, 0.1, 1, 10$. Table 3 shows the experimental results for $\alpha = 0.1$. The results for $\alpha = 0.01, 1, 10$ were similar. We observed that the number of clusters increases along with the number of features for $m \leq 500$; for $m > 500$ the number of clusters decreases significantly. Compared to structural clustering, DP clustering produces less clusters. In order to make the results more comparable to the results of our method, we varied the user-specified parameters. Nonetheless, it is impossible to parameterize the DP clustering method to obtain more than seven clusters. Figure 5(b) presents the clustering results for $m = 1000$, since Tsuda reported a good behavior of the algorithm for this value. Moreover, additional features can reveal detailed structure of the data. However, this advantage presents a disadvantage at the same time, since graph clusters with thousands of features are difficult to interpret. The DP clustering results indicate that the method is not able to discriminate the known structure classes in the SACA data set very well. In contrast to the results of structural clustering presented in Section 3, the DP clusters are, in many cases, associated with different structure classes, indicated by lower values of the Rand index (Table 3).

Table 3. Number of clusters and size of the DFS code tree for DP clustering on the SACA data set with $\alpha = 0.1$

# Features	50	100	500	1000	5000
# Clusters	6	7	7	6	2
Rand Index	0.639	0.572	0.761	0.747	0.364

Cluster Stability. The outcome of the proposed structure-based clustering approach is dependent on the order in which the objects in the data set are processed. In this experiment, we studied the impact of the order of objects in the data sets by assessing the stability of the resulting clusters. We performed several experiments on data sets that are based on different permutations of a data set. The Tanimoto similarity coefficient was used as a cluster-wise measure of cluster stability, which defines the similarity between two clusters in terms of the intersection of common instances compared to the union of common instances. Our results suggest that structural clustering is sensitive to the particular order of the data sets. However, the obtained clusters are stable with respect to data permutations, since approximately 85% of the clusters of size ≥ 2 of a clustering yield a Tanimoto similarity value of 1 compared to the most similar cluster in a reference clustering. Taking also singletons into account, the similarity of the clusters rises to 94%.

Performance with/without Cluster Exclusion Criteria. We investigated the impact of the cluster exclusion criteria defined in Equation 3 and 4 on the performance of the structure-based clustering algorithm. Therefore, we ran the algorithm on the data sets in Table 1 with and without the exclusion criteria. Figure 6 shows the results of the experiment on three representative data sets, i.e. CPDBAS_MOUSE, CPDBAS_RAT and EPAFHM. The results indicate that a significant performance improvement can be achieved with the application of the cluster exclusion criteria.

Scalability Experiments. To study the scalability, we performed experiments on ten data sets from the NCI anti-HIV database that consist of x graphs ($x \in [1000, 10000]$) with a similarity coefficient $\theta \in [0.2, 0.8]$. As it can be seen in Figure 7, the structure-based clustering algorithm scales favorably as the size of

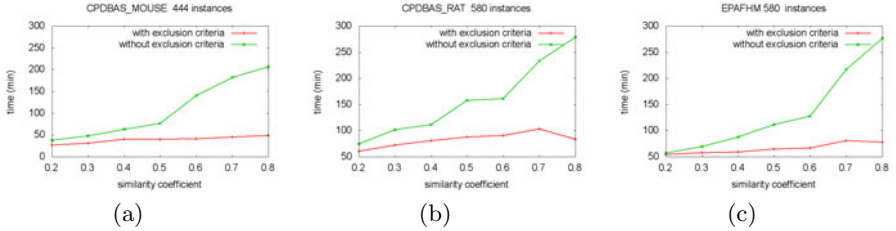


Fig. 6. Runtime performance of structure-based clustering with and without clustering exclusion criteria on (a) CPDBAS_MOUSE, (b) CPDBAS_RAT and (c) EPAFHM

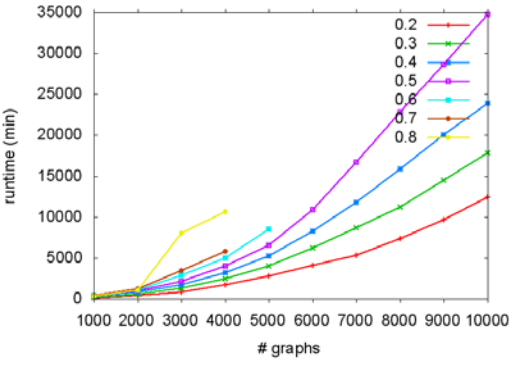


Fig. 7. Runtime performance of the structure-based clustering approach on ten data sets from the NCI anti-HIV database consisting of x graphs ($x \in [1000, 10000]$)

the data set increases. However, for $0.6 \leq \theta \leq 0.8$, the algorithm did not respond within a certain timeout period for a data set size larger than 4000 / 5000 objects. The overall results suggest that, depending on reasonable parameter settings, our clustering approach can handle data sets of at least 10,000 graphs.

4 Related Work

Graph clustering has been extensively investigated over the past few years. Basically, there exist two complementary approaches to graph clustering [9]. The simpler and more established one is to calculate a vectorial representation of the graphs and use standard similarity or distance measures in combination with standard clustering algorithms. The feature vector can be composed of properties of the graph and / or of subgraph occurrences [10]. Yoshida *et al.* [11] describe a graph clustering method based on structural similarity of fragments (connected subgraphs are considered) in graph-structured data. The approach is experimentally evaluated on synthetic data only and does not consider edge and node labels. The second approach is to use the structure of the graphs directly. Tsuda and Kudo [3] proposed an EM-based method for clustering graphs based on weighted frequent pattern mining. Their method is fully probabilistic, adopting a binomial mixture model defined on a very high dimensional vector indicating the presence or absence of all possible patterns. However, the number of clusters has to be specified a priori, and the model selection procedure is not discussed in their paper. Bunke *et al.* [12] proposed a new graph clustering algorithm, which is an extension of Kohonen's well-known Self-Organizing Map (SOM) algorithm [13] into the domain of graphs. The approach is experimentally evaluated on the graph representations of capital letters that are composed of straight line segments. Chen and Hu [14] introduced a non-exhaustive clustering algorithm that allows for clusters to be overlapping by modifying the traditional k-medoid algorithm. This implies the drawback that the number of clusters has

to be known beforehand. In a comparison of an MCS-based clustering with a fingerprint-based clustering, Raymond *et al.* [15] report that no obvious advantage results from the use of the more sophisticated, graph-based similarity measures. They draw the conclusion, that although the results obtained from the use of graph-based similarities are different from fingerprint-based similarities, there is no evidence to suggest that one approach is consistently better than the other.

Summing up, all MCS-based clustering approaches appear to suffer from the NP-hardness of the MCS computation. While no running times are reported and no implementations appear to be publicly available, the graph data sets tested in related papers typically contain less than 500 graphs [4, 15]. Our choice to modify a frequent graph miner instead is partly motivated by the observation by Bunke *et al.* [16] that for dense graphs association graph methods are preferable, whereas for sparse graphs (as the molecular structures occurring in our application domains) methods enumerating frequent subgraphs are to be preferred.

Related to our clustering approach, Aggarwal *et al.* [17] propose a structural clustering method for clustering XML data. It employs a projection based structural approach and uses a set of frequent substructures as the representative with respect to an intermediate cluster of XML documents. Paths extracted from XML documents are used as a document representation. To mine frequent closed sequences, the sequential pattern mining algorithm BIDE [18] was revised in order to terminate search once a sequence reaches a specified size. Our work is most closely related to the graph clustering approach by Tsuda and Kurihara [5]. They presented a graph clustering approach based on frequent pattern mining that addresses the problem of learning a DP mixture model in the high dimensional feature space of graph data. To keep the feature search space small, an effective tree pruning condition was designed. Although our clustering approach is similarly based on frequent subgraph mining there are several important differences. First, there are differences in the output that makes our clustering results easier to interpret. Despite the proposed feature selection method to obtain a reduced feature set, DP clustering still outputs quite numerous frequent subgraphs which make the graph clusters difficult to interpret. In contrast, our method actually outputs just the clustered graphs sharing a common cluster scaffold. Second, we provide an effective online algorithm that allows for overlapping and non-exhaustive clustering. Non-exhaustive clustering may be more robust in case the set of objects to be clustered contains outliers. The rationale of overlapping clustering is that in some applications it is not appropriate for an object to belong to only one cluster.

5 Conclusion and Future Work

We presented a new online algorithm for clustering graph objects in terms of structural similarity. Structural graph clustering can offer interesting new insights into the composition of graph data sets. Moreover, it can be practically useful to benchmark other graph mining algorithms, to derive new substructural

descriptors, to compute local models for classifying graphs, and to calculate the applicability domain of models. Several experiments were designed to evaluate the effectiveness and efficiency of our approach on various real world data sets of molecular graphs. First of all, the results indicate that the clustering method is able to rediscover known structure classes in the NCI standard anti-cancer agents. Moreover, a baseline comparison with a fingerprint-based clustering was presented. The results demonstrate that the structural clustering approach yields larger and more representative cluster scaffolds compared to FP based clustering, thus reducing the heterogeneity in the clusters obtained by fingerprint clustering. To show the importance of the cluster exclusion criteria defined in Equation 3 and 4, we evaluated the performance of the structural clustering approach with and without these criteria. Finally, to investigate how well the algorithm scales regarding running time, we performed extensive experiments with 10,000 compounds selected from the NCI aids anti-viral screen data. In summary, our results suggest that this overlapping, non-exhaustive structural clustering approach generates interpretable clusterings in acceptable time. Further work, from an application point of view, includes the following: First, it would be interesting to investigate the effects of preprocessing steps, e.g., down-weighting longer chains (acyclic substructures) or reduced graph representations (transforming cycles, in chemical terms: rings, into special nodes). Second, the algorithm could be extended easily to take into account the physico-chemical properties of whole molecules. Technically, this would mean that only graphs within a certain distance with respect to such global graph properties are added to a cluster.

Acknowledgements

This work was partially supported by the German Federal Ministry for the Environment, Nature Conservation and Nuclear Safety and the EU FP7 project (HEALTH-F5-2008-200787) OpenTox (http://www.opentox.org).

References

1. Inokuchi, A., Washio, T., Motoda, H.: An APriori-based algorithm for mining frequent substructures from graph data. In: PKDD '00: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery, pp. 13–23 (2000)
2. Yan, X., Han, J.: gSpan: Graph-based substructure pattern mining. In: Proceedings of the 2002 IEEE International Conference on Data Mining, pp. 721–724 (2002)
3. Tsuda, K., Kudo, T.: Clustering Graphs by Weighted Substructure Mining. In: Cohen, W.W., Moore, A. (eds.) ICML 2006, pp. 953–960. ACM Press, New York (2006)
4. Stahl, M., Mauser, H.: Database clustering with a combination of fingerprint and maximum common substructure methods. J. Chem. Inf. Model. 45, 542–548 (2005)
5. Tsuda, K., Kurihara, K.: Graph mining with variational Dirichlet process mixture models. In: Proceedings of the 8th SIAM International Conference on Data Mining, pp. 432–442 (2008)

6. Martin, Y.C., Kofron, J.L., Traphagen, L.M.: Do structurally similar molecules have similar biological activity? J. Med. Chem. 45, 4350–4358 (2002)
7. Weinstein, J., Kohn, K., Grever, M., Viswanadhan, V.: Neural computing in cancer drug development: Predicting mechanism of action. Science 258, 447–451 (1992)
8. Koutsoukos, A.D., Rubinstein, L.V., Faraggi, D., Simon, R.M., Kalyandrug, S., Weinstein, J.N., Kohn, K.W., Paull, K.D.: Discrimination techniques applied to the NCI in vitro anti-tumour drug screen: predicting biochemical mechanism of action. Stat. Med. 13, 719–730 (1994)
9. Raymond, J.W., Willett, P.: Effectiveness of graph-based and fingerprint-based similarity measures for virtual screening of 2D chemical structure databases. J. Comput. Aided. Mol. Des. 16(1), 59–71 (2002)
10. McGregor, M.J., Pallai, P.V.: Clustering of large databases of compounds: Using the MDL "keys" as structural descriptors. J. Chem. Inform. Comput. Sci. 37(3), 443–448 (1997)
11. Yoshida, T., Shoda, R., Motoda, H.: Graph clustering based on structural similarity of fragments. In: Jantke, K.P., et al. (eds.) Federation over the Web. LNCS (LNAI), vol. 3847, pp. 97–114. Springer, Heidelberg (2006)
12. Günter, S., Bunke, H.: Validation indices for graph clustering. Pattern Recogn. Lett. 24(8), 1107–1113 (2003)
13. Kohonen, T.: Self-organizing maps. Springer, Heidelberg (1997)
14. Chen, Y.L., Hu, H.L.: An overlapping cluster algorithm to provide non-exhaustive clustering. Eur. J. Oper. Res. 173(3), 762–780 (2006)
15. Raymond, J.W., Blankley, C.J., Willett, P.: Comparison of chemical clustering methods using graph- and fingerprint-based similarity measures. J. Mol. Graph. Model. 21(5), 421–433 (2003)
16. Bunke, H., Foggia, P., Guidobaldi, C., Sansone, C., Vento, M.: A comparison of algorithms for maximum common subgraph on randomly connected graphs. In: Caelli, T.M., Amin, A., Duin, R.P.W., Kamel, M.S., de Ridder, D. (eds.) SPR 2002 and SSPR 2002. LNCS, vol. 2396, pp. 123–132. Springer, Heidelberg (2002)
17. Aggarwal, C.C., Ta, N., Wang, J., Feng, J., Zaki, M.: XProj: a framework for projected structural clustering of XML documents. In: KDD '07: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 46–55. ACM, New York (2007)
18. Wang, K., Han, J.: Bide: Efficient mining of frequent closed sequences. In: International Conference on Data Engineering (2004)