Suppression Distance Computation for Set Covers and Hierarchies

François Queyroi, Sergey Kirgizov

Suppression Distance Computation for Set Covers and Hierarchies

François Queyroi*, Sergey Kirgizov*

Sorbonne Universités, UPMC Univ Paris 06, UMR 7606, LIP6, F-75005
CNRS, UMR 7606, LIP6, F-75005, Paris, France

Abstract

We discuss the computation of the suppression distance between two set covers. It is the minimum number of element of a set that have to be removed so the renaming covers are equal. We show that this problem is $\mathcal{NP}$-Hard by reduction to MINIMUM VERTEX COVER. We further investigate the subclass of hierarchies and prove this problem to be polynomial in this case as it gives birth to a class of perfect graphs. We also propose an algorithm based on recursive maximum assignments.

Keywords: set cover, hierarchy, distance, graphs, vertex cover

1. Introduction

Decomposing a set into patterns of interest is a central problem in data analysis. Evaluating the distance between decompositions is an important task in this context as it allows to study the behaviour of clustering algorithms or study the evolution of a set of patterns over time. The situation where the detected patterns do not overlap is called *partitions*. Measures based on edit distance [4, 9] or on mutual information [8] have been used to assess the distance between partitions. The situation where groups may overlaps received some attention recently [3]: we call these decomposition *covers*. They are particularly useful for the detection of communities in complex networks [6]. A subclass of covers are *hierarchies* in which patterns can

*Corresponding author
Email addresses: francois.queyroi@lip6.fr (François Queyroi), sergey.kirgizov@lip6.fr (Sergey Kirgizov)

be recursively decomposed into smaller patterns with similar properties. The problem of distance definition between hierarchies is also of interest [10]. We focus in this paper on the computation of the *suppression distance* defined for partitions in [4] and generalized here for general covers and hierarchies.

We assume we have a set S of elements of finite cardinality n. A *cover* of the set S is a finite collection of subsets $\mathcal{C} = (C_1, C_2, \ldots, C_k)$ where each C_i is a non-empty subset of S. We call $\mathcal{C}(S')$ the sub-collection of groups that contain $S' \subseteq S$. We denote by $\mathcal{C}[S']$ the *induced sub-cover* of $\mathcal{C}$ by S *i.e.* the cover of S' obtained after the removal of every elements of $S \setminus S'$.

Definition 1. ***(Suppression Distance)*** *Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two covers of S. The suppression distance d_s is defined as*

$$d_s(\mathcal{C}_1, \mathcal{C}_2) = \min_{S' \subseteq S}\{|S'|, \mathcal{C}_1[S \setminus S'] = \mathcal{C}_2[S \setminus S']\} \tag{1}$$

A set S' such that $\mathcal{C}_1[S \setminus S'] = \mathcal{C}_2[S \setminus S']$ is called a suppression set.

Theorem 1. *The function d_s is a metric.*

Proof. The non-negativity, identity of indiscernibles and symmetry properties are straightfoward for d_s. Moreover, this distance respects the triangular inequality. Consider three covers $\mathcal{C}_1, \mathcal{C}_2$ and $\mathcal{C}_3$. Let S_{ij} be such set that $\mathcal{C}_i[S \setminus S_{ij}] = \mathcal{C}_j[S \setminus S_{ij}]$. Observe that $\mathcal{C}_1[S \setminus (S_{12} \cup S_{23})] = \mathcal{C}_3[S \setminus (S_{12} \cup S_{23})]$, so we have:

$$\begin{aligned} |S_{13}| &\leq |S_{12} \cup S_{23}| \leq |S_{12}| + |S_{23}| \\ d_s(\mathcal{C}_1, \mathcal{C}_3) &\leq d_s(\mathcal{C}_1, \mathcal{C}_2) + d_s(\mathcal{C}_2, \mathcal{C}_3) \end{aligned}$$

□

2. Computation of the Suppression Distance

The evaluation of the distance between covers relates to *hypergraph matching* used in pattern recognition [2]. One can easily reduce the computation of the suppression distance to the problem of MAXIMUM COMMOM SUB-HYPERGRAPH [1]. Indeed the couple $(S, \mathcal{C})$ is an *hypergraph.* If $S' \subset S$ is the maximum common induced sub-hypergraph of $(S, \mathcal{C}_1)$ and $(S, \mathcal{C}_2)$ then obviously we have $d_s(\mathcal{C}_1, \mathcal{C}_2) = |S| - |S'|$. However, finding such subset is

$\mathcal{NP}$-hard. We show here that the suppression distance computation can also be reduced to the MINIMUM VERTEX COVER problem. The differences between both covers can be encoded into a simple graph called the *cover graph.*

Definition 2. ***(Cover Graph)*** *Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two covers of a set S. We call $G(\mathcal{C}_1, \mathcal{C}_2) = (S, E)$ the* cover graph *of $(\mathcal{C}_1, \mathcal{C}_2)$ with the edges $E = \{(s_1, s_2) \in S^2, |\mathcal{C}_1(\{s_1, s_2\})| \neq |\mathcal{C}_2(\{s_1, s_2\})|\}$. This graph can contain self-loops.*

Two elements (or a singleton) of S are connected iff they (it) do (does) not appear the same amount of time together (alone) in both covers. A similar transformation has been proposed by Gusfield [4].

Lemma 1. *Given $G = (S, E)$ the cover graph of $(\mathcal{C}_1, \mathcal{C}_2)$ and $S' \subseteq S$, the induced subgraph $G[S']$ is the cover graph of $(\mathcal{C}_1[S'], \mathcal{C}_2[S'])$.*

Proof. First, the $G(\mathcal{C}_1[S'], \mathcal{C}_2[S'])$ has the same vertex set as $G[S']$. Second, for every pair of elements s_1, s_2 in S', we have $|\mathcal{C}_1(\{s_1, s_2\})| = |\mathcal{C}'_1(\{s_1, s_2\})|$ and $|\mathcal{C}_2(\{s_1, s_2\})| = |\mathcal{C}'_2(\{s_1, s_2\})|$. Therefore, every edge in $G(\mathcal{C}_1[S'], \mathcal{C}_2[S'])$ is also an edge in $G[S']$ since $G[S']$ is an induced subgraph. Also, every non-edge in $G(\mathcal{C}_1[S'], \mathcal{C}_2[S'])$ is an non-edge in $G[S']$. Therefore the edge set of both graphs are equal. □

Theorem 2. *$d_s(\mathcal{C}_1, \mathcal{C}_2)$ is equal to the size of the minimum vertex cover of $G(\mathcal{C}_1, \mathcal{C}_2)$.*

Proof. We first show that $E = \emptyset$ iff $\mathcal{C}_1 = \mathcal{C}_2$. If both covers are equal then each pair of vertices appears the same amount of time in both decomposition and the cover graph contains no edge. Now if a cover graph contains no edge then there exist a bijection between the groups of $\mathcal{C}_1$ and $\mathcal{C}_2$. If such bijection does not exist, it means there is at least one group in either $\mathcal{C}_1$ or $\mathcal{C}_2$ that has no equal counterpart in the other cover. Therefore, there is a pair of elements in S^2 that does not appear together the same amount of time in $\mathcal{C}_1$ and $\mathcal{C}_2$. Those two elements should be connected in G which contradicts our hypothesis. Finding a set S' such that $\mathcal{C}_1[S \setminus S']$ and $\mathcal{C}_2[S \setminus S']$ are equal is therefore equivalent to finding a S' such that $G[S \setminus S']$ contain no edges (Lemma 1). If the S' is the smallest among all other subsets with this property then it is a minimum vertex cover of G. □

Theorem 3. *For any simple graph $G = (S, E)$, there exist two covers $(\mathcal{C}_1, \mathcal{C}_2)$ of S such that $G = G(\mathcal{C}_1, \mathcal{C}_2)$*

Proof. For each edge $(u, v) \in E(G)$, add the set $\{u, v\}$ to $\mathcal{C}_1$ and create two groups $\{u\}$ and $\{v\}$ in $\mathcal{C}_2$. Using this construction, the elements u and v are found together in one group in $\mathcal{C}_1$ and are not found together in $\mathcal{C}_2$. Therefore the cover graph $G(\mathcal{C}_1, \mathcal{C}_2)$ will contain the edge (u, v). Moreover, the edge set of $G(\mathcal{C}_1, \mathcal{C}_2)$ will correspond to the groups of $\mathcal{C}_1$. We conclude $G = G(\mathcal{C}_1, \mathcal{C}_2)$. □

Since any graph can be the cover graph of two set covers (Theorem 3), computing $d_s(\mathcal{C}_1, \mathcal{C}_2)$ is $\mathcal{NP}$-hard by reduction to the Minimum Vertex Cover problem (Theorem 2).

3. The Case of Hierarchies

Hierarchies are a particular class of covers that contain partitions as special cases. We show in this section that d_s can be computed in polynomial time in this context and provide a recursive algorithm.

Definition 3. ***(Hierarchy)*** *A hierarchy $\mathcal{H}$ is a cover of a set S such that if there exist two groups $H_1, H_2 \in \mathcal{H}$ such that $H_1 \cap H_2 \neq \emptyset$ then either $H_1 \subseteq H_2$ or $H_2 \subseteq H_1$.*

Since the inclusion between the sets define a weak ordering of the groups. The relations between the sets of $\mathcal{H}$ can be represented in a tree ordered fashion, the roots being the groups that are not include in any other group. Let $N_i(\mathcal{H})$ denote the i-th level of $\mathcal{H}$ *i.e.* the groups sitting at depth i in this tree. Notice it is still well defined if $\mathcal{H}$ contains repeated groups. The *depth* of a hierarchy is the maximal depth of its groups. For $G = G(\mathcal{H}_1, \mathcal{H}_2)$, we have a function $p : E(G) \to \mathbb{N}$ which is the first level at which the couple $(a, b) \in E$ belongs to a group of $\mathcal{H}_1$ but not $\mathcal{H}_2$ (or the opposite). We denote by G_i the subgraph of G formed by the edges $\{e \in E, p(e) \geq i\}$. Notice we have $G_1 = G$. An example of hierarchies and their cover graph can be found in Figure 1.

We will assume that each element of S appears the same number of times in $(\mathcal{H}_1, \mathcal{H}_2)$. If it not the case, the cover graph $G(\mathcal{H}_1, \mathcal{H}_2)$ contains self-loops. A subset of vertices belonging to all minimum vertex covers is therefore straightforward to find. Indeed, vertices with self-loops are part of all minimum vertex cover.

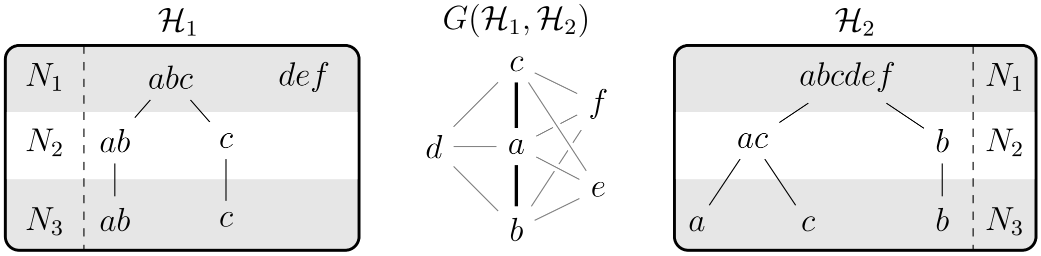

Figure 1: Example of two hierarchies $\mathcal{H}_1, \mathcal{H}_2$ of a set $S = \{a, b, c, d, e, f\}$ and their cover graph $G(\mathcal{H}_1, \mathcal{H}_2)$ ($p(e) = 1$ for gray edges and $p(e) = 2$ for black edges).

3.1. Existence of a polynomial-time solution

We show here that the cover graph of hierarchies is a perfect graph. As said before, we assume that each element of S belongs to the same number of groups in both hierarchies. One important consequence of this is the fact that $\mathcal{H}_1, \mathcal{H}_2$ have the same depth d.

Lemma 2. *Let $S' \subseteq S$ and $i > 0$ such that $G_i[S']$ is connected, the elements of S' all belong to the same groups of depth lower than i in both $\mathcal{H}_1$ and $\mathcal{H}_2$.*

Proof. Assume $G_i[S']$ is a connected subgraph but there exist at least two non-overlapping subsets A and B of S' that belong to different groups of depth lower than i in both $\mathcal{H}_1$ and $\mathcal{H}_2$. It means that either A and B belong to the same groups in $\mathcal{H}_1$ but not in $\mathcal{H}_2$ or both do not belong to the same groups in both hierarchies. Notice that both cases are impossible otherwise all edges would have value a value lower than i (first case) or there would be no edges between the two groups (second case). This contradicts our hypothesis since we assume $A \cup B$ is a connected component of G_i. □

Lemma 3. *Let $S' \subseteq S$ and $i > 0$ such that $G_i[S']$ is connected, if there exist $u \notin S'$ such that $(u, v) \in E$ and $p(u, v) = j < i$ then $\forall w \in S'$, we have $(u, w) \in E$ with $p(u, w) = j$.*

Proof. According to Lemma 2, the elements in S' all belongs to the same groups of depth lower than i in both hierarchies. If there exist $u \notin S'$ such that $(u, v) \in E$ and $p(u, v) = j < i$, it means there is a group at depth j in $\mathcal{H}_1$ that contain $(u \cup S')$ and a group at depth j in $\mathcal{H}_2$ that contains S' but not u. Therefore u should be connected to every elements of S' with an edge of value j. □

One important consequence of Lemma 3 is that two connected components of G_2 either have no edges between them or form a complete bipartite subgraph in G.

Theorem 4. *Let $\mathcal{H}_1, \mathcal{H}_1$ be two hierarchies of finite depth of a set S, computing $d_s(\mathcal{H}_1, \mathcal{H}_1)$ can de done in polynomial time.*

Proof. Let Ψ_d denote the set of hierarchies of a set S of depth d, we show that the cover graph G (obtained after the removal of vertices with self-loops) of any pair in Ψ_d is a perfect graph by induction over d.

For $d = 1$, the class Ψ_1 corresponds to the class of partitions of S and the graph G is therefore perfect (Theorem 3.4 of [4]).

Assume it true for any d we show it is also true for $d + 1$. Let $\tilde{G}$ be the graph obtained by contraction of edges of G_2 in G. This graph is perfect. Indeed, according to Lemma 2, elements within the same connected components of G_2 belongs to the same group at depth 1. The graph $\tilde{G}$ is therefore the cover graph of the two flat partitions $(\tilde{\mathcal{P}_1}, \tilde{\mathcal{P}_2})$ obtained by fusioning each group into a single new element in the partitions $N_1(\mathcal{H}_1)$ and $N_1(\mathcal{H}_2)$.

According to Lemma 3, the graph G can be recovered from $\tilde{G}$ by expanding each vertex $u \in V(\tilde{G})$ by its corresponding connected component of G_2 and connecting each element to the vertices previously adjacent to u. Now, since every connected subgraphs of G_2 is perfect as the cover graph of two hierarchies of depth d, the graph G is also perfect according to the Theorem 1 of [7]. A minimum vertex cover of G can be found in polynomial time. □

3.2. A solution based on recursive maximum assignment

It has been shown that, if $\mathcal{P}_1$ and $\mathcal{P}_2$ are partitions, the distance $d_s(\mathcal{P}_1, \mathcal{P}_2)$ can be computed by solving a MAXIMUM ASSIGNMENT problem based on the size of intersections between all pairs of groups in $\mathcal{P}_1$ and $\mathcal{P}_2$ [4]. The computation of the intersections takes $\mathcal{O}(n)$ and the assignment is computed in $\mathcal{O}((|\mathcal{P}_1| + |\mathcal{P}_2|)^3)$. The set of elements to be removed are the elements in the sets that are not covered by the maximum assignment.

Notice two hierarchies are equal if all their levels are equal *i.e.* if the set of couple $\{(N_i(\mathcal{H}_1), N_i(\mathcal{H}_2))\}_{1 \leq i \leq d}$ are all pairwise equal. However, finding a suppression set S' using a greedy "level-by-level" approach (either top-down or bottom-up) may not lead to a optimal solution. Consider the example given in Figure 1 where $d_s(\mathcal{H}_1, \mathcal{H}_2) = 3$, a top-down approach may fail since either $\{a, b, c\}$ or $\{d, e, f\}$ can be chosen at level 1 to be part of S'. But choosing $\{d, e, f\}$ would lead to a distance of 4. Alternatively, consider the sub-hierarchies induced by the set $\{a, b, c\}$, a bottom-top approach may also fail since either a or b can belongs to S' at the last level. Choosing b would lead to a distance of 2 whereas $d_s(\mathcal{H}_1[\{a, b, c\}], \mathcal{H}_2[\{a, b, c\}]) = 1$.

We provide here an algorithm for computing a suppression set for two hierarchies and therefore the distance between them. The idea is to recursively compute a suppression set for two sub-hierarchies whose element belongs to the same groups at the current level.

```
Algorithm 1: suppressionSet($\mathcal{H}_1, \mathcal{H}_2$)
Input: $\mathcal{H}_1, \mathcal{H}_2$ two hierarchies of a set $S$
Output: $S' \subseteq S$ a suppression set
if $\mathcal{H}_1 = \mathcal{H}_2 = \emptyset$ then
    return $\emptyset$
else
    $S' \leftarrow \emptyset$
    for $C \in maxCommonGroups(N_1(\mathcal{H}_1), N_1(\mathcal{H}_2))$ do
        $S' \leftarrow S' \cup suppressionSet(\mathcal{H}_1[C] - C, \mathcal{H}_2[C] - C)$
    end
    return $S' \cup flatSuppressionSet(N_1(\mathcal{H}_1[S \setminus S']), N_1(\mathcal{H}_2[S \setminus S']))$
end
```

The function *maxCommonGroups*(P_1, P_2) returns the maximal subsets of vertices that are together in both partitions P_1 and P_2. The function *flatSuppressionSet*(P_1, P_2) returns a minimum suppression set of elements S' such that $d_s(P_1, P_2) = |S'|$. This function can used the Hungarian algorithm [5]. The following result is used to show that the set returned by a recursion call is a subset of one optimal solution.

Lemma 4. *Let $G = (S, E)$ be the cover graph of two hierarchies of S, any minimum vertex cover of G_i is a subset of a minimum vertex cover of G_{i-1}.*

Proof. Let C be a minimum vertex cover of G_i, S' be a maximal connected component of G_i. The set $C' = S' \cap C$ is a minimum vertex cover of $G_{i-1}[S']$. According to Lemma 3, the edge cut (S', S'') forms a complete bipartite graph where S'' is the set of vertices in $(S \setminus S')$ connected to S'. Therefore, the minimum vertex cover of G_{i-1} should contain either all S' or all $S'' \cup C'$ *i.e.* C' is a subset of the cover in both cases. Since it is true for the minimum cover of every maximal connected components of G_i, the set C is a subset a minimum vertex cover of G_{i-1}. □

Theorem 5. *For two hierarchies $\mathcal{H}_1, \mathcal{H}_2$ of a set S, Algorithm 1 always terminates and returns a suppression set S' such that $d_s(\mathcal{H}_1, \mathcal{H}_2) = |S'|$.*

Proof. Termination: First the algorithm will terminate since the hierarchies are of finite depth and the recursive call is used on two sub hierarchies of lower depth (the "root" group is removed in both hierarchies in line 6). Moreover, the condition in line 1 is always met at some point since we assume elements of S appears the same number of times in both hierarchies.

Correctness: Observe the function *maxCommonGroups*(P_1, P_2) returns sets that correspond to either independent or maximal connected components of G_2. Let C be one of these sets, the sub-hierarchies $\mathcal{H}_1[C]-C$ and $\mathcal{H}_2[C]-C$ correspond to the sub-hierarchies induced by C minus the first level. Assume that at the end of the loop 5–7, the set S' is the union of the elements to be removed so that those sub-hierarchies are equal. According to Lemma 4, the set S' is a subset of a minimum suppression set between $(\mathcal{H}_1, \mathcal{H}_2)$ *i.e.* a minimum vertex cover of G_2. A possible solution is therefore the union of S' and a suppression set of $\mathcal{H}_1[S \setminus S']$ and $\mathcal{H}_2[S \setminus S']$. The latter can be found only looking at the first level of both hierarchies. We can show the assumption on S' to be true by induction since the Algorithm will return a minimum suppression set if $\mathcal{H}_1, \mathcal{H}_2$ are flat partitions. □

We shall discuss the complexity of Algorithm 1. The function *maxCommonGroups* can be implemented in $\mathcal{O}(n \log(n))$. Notice the worst case scenario is achieved when $\mathcal{H}_1 = \mathcal{H}_2$ since the results of each recursive call will be the empty set. Assuming the first $\Theta(d)$ levels correspond to one repeated group S and the last levels correspond to $\Theta(n)$ repeated groups, the time complexity is $\mathcal{O}(n^3 + dn \log n)$.

References

[1] Bunke, H., 1997. On a relation between graph edit distance and maximum common subgraph. Pattern Recognition Letters 18 (8), 689–694.

[2] Bunke, H., Dickinson, P., Kraetzl, M., Neuhaus, M., Stettler, M., 2008. Matching of hypergraphs—algorithms, applications, and experiments. In: Applied Pattern Recognition. Springer, pp. 131–154.

[3] Goldberg, M. K., Hayvanovych, M., Magdon-Ismail, M., 2010. Measuring similarity between sets of overlapping clusters. In: Social Computing (SocialCom), 2010 IEEE Second International Conference on. IEEE, pp. 303–308.

[4] Gusfield, D., 2002. Partition-distance: A problem and class of perfect graphs arising in clustering. Information Processing Letters 82 (3), 159–164.

[5] Kuhn, H. W., 1955. The hungarian method for the assignment problem. Naval research logistics quarterly 2 (1-2), 83–97.

[6] Lancichinetti, A., Fortunato, S., Kertész, J., 2009. Detecting the overlapping and hierarchical community structure in complex networks. New Journal of Physics 11 (3), 033015.

[7] Lovász, L., 1972. Normal hypergraphs and the perfect graph conjecture. Discrete Mathematics 2 (3), 253–267.

[8] Meilă, M., 2003. Comparing clusterings by the variation of information. In: Learning theory and kernel machines. Springer, pp. 173–187.

[9] Porumbel, D. C., Hao, J. K., Kuntz, P., 2011. An efficient algorithm for computing the distance between close partitions. Discrete Applied Mathematics 159 (1), 53–59.

[10] Queyroi, F., Delest, M., Fédou, J.-M., Melançon, G., 2014. Assessing the quality of multilevel graph clustering. Data Mining and Knowledge Discovery, 1–22.