Clustering Complex Data Represented as propositional formulas

Abdelhamid Boudane, Said Jabbour, Lakhdar Saïs, Yakoub Salhi

Clustering Complex Data Represented as propositional formulas

Abdelhamid Boudane, Said Jabbour, Lakhdar Sais, and Yakoub Salhi

CRIL-CNRS, Université d'Artois, F-62307 Lens Cedex, France
{boudane,jabbour,sais,salhi}@cril.fr

Abstract. Clustering has been extensively studied to deal with different kinds of data. Usually, datasets are represented as a n-dimensional vector of attributes described by numerical or nominal categorical values. Symbolic data is another concept where the objects are more complex such as intervals, multi-categorical or modal. However, new applications might give rise to even more complex data describing for example customer desires, constraints, and preferences. Such data can be expressed more compactly using logic-based representations. In this paper, we introduce a new clustering framework, where complex objects are described by propositional formulas. First, we extend the two well-known k-means and hierarchical agglomerative clustering techniques. Second, we introduce a new divisive algorithm for clustering objects represented explicitly by sets of models. Finally, we propose a propositional satisfiability based encoding of the problem of clustering propositional formulas without the need for an explicit representation of their models. Preliminary experimental results validating our proposed framework are provided.

1 Introduction

Clustering is a technique used to recover hidden structure in a dataset obtained by grouping data into clusters of similar objects. It is derived by several important applications ranging from scientific data exploration, to information retrieval, and computational biology (e.g. [1]). Such diversity in terms of application domains induces a variety of data types and clustering techniques (see. [2] for a survey). Indeed, data can be transactional, sequential, trees, graphs, texts, or even of a symbolic nature [6, 7, 10]. This last kind of data is particularly suitable for modeling complex and heterogeneous objects usually described by a set of multivalued variables of different types (e.g. intervals, multi-categorical or modal) (e.g. [3, 4, 8]). We can also mention conceptual clustering proposed more than thirty years ago by Michalski [14] and defined as a machine learning task. It accepts a set of object descriptions (events, facts, observations, ...) and produces a classification scheme over them. Conceptual clustering not only partitions the data, but generates clusters that can be summarized by a conceptual description. As a summary, conceptual and symbolic clustering are twoparadigm proposed to deal with kinds of data other than those usually described by numerical values.

In today's data-driven digital era, data might be even more complex and heterogenous. Such complex data might represent customers desires or preferences

collected in different possible ways using surveys and quizzes. As an example, one can cite configuration systems usually designed to provide customized products satisfying the different requirements of the customer, usually modeled by constraints or logic-formulas (e.g. [11]). These customers requirements-data or the data-models provided by the configuration systems are some kind of complex data that we are interested in. These data can be represented by logic-formulas (requirements) or by models (the products satisfying the requirements). Data can also represent more complex entities such as transaction databases. Indeed, suppose that we collected several transaction databases from stores chain selling the same products, one can be interested in determining similar stores (clusters) or stores with the same behavior. This could help the manager of the stores chain to better define its trade policy. In the two previous examples, data can be better represented as a set of propositional formulas or as sets of models.

In this paper, we introduce a new clustering framework, where complex objects are described by propositional formulas. We first extend the two well known k-means and hierarchical agglomerative clustering techniques. Then, we introduce a new divisive algorithm for clustering objects represented explicitly by sets of models. Finally, we propose a propositional satisfiability based encoding of clustering propositional formulas without the need for an explicit representation of their models. Preliminary experimental results validating our proposed framework are provided before concluding.

1.1 Propositional Satisfiability

Let $\mathcal{P}$ be a countably infinite set of propositional variables. The set of *propositional formulas*, denoted $F_{\mathcal{P}}$, is defined inductively starting from $\mathcal{P}$, the constant $\bot$ denoting absurdity, the constant $\top$ denoting true, We use the greek letters ϕ, ψ to represent formulas. A *Boolean interpretation* $\mathcal{I}$ of a formula ϕ is defined as a function from $\mathcal{P}(\phi)$ to $\{0,1\}$ (0 for $false$ and 1 for $true$). A *model* of a formula ϕ is a Boolean interpretation $\mathcal{I}$ that satisfies ϕ (written $\mathcal{I} \vDash \phi$), i.e. $\mathcal{I}(\phi) = 1$. We denote the set of models of ϕ by $\mathcal{M}(\phi)$. A formula ϕ is satisfiable (or consistent) if there exists a model of ϕ; otherwise it is called unsatisfiable (or inconsistent).

Let ϕ and ψ be two propositional formulas, we say that ψ is a logical consequence of ϕ, written $\phi \vDash \psi$, iff $\mathcal{M}(\phi) \subseteq \mathcal{M}(\psi)$. The two formulas ϕ and ψ are called equivalent iff $\phi \vDash \psi$ and $\psi \vDash \phi$, i.e $\mathcal{M}(\phi) = \mathcal{M}(\psi)$.

A CNF formula is a conjunction ($\wedge$) of clauses, where a *clause* is a disjunction ($\vee$) of literals. A *literal* is a propositional variable (p), called positive literal, or ($\neg p$), called negative literal. The *SAT problem* consists in deciding whether a given CNF formula admits a model or not. Another problem related to SAT is the SAT model enumeration problem. Enumeration requires generating all models of a problem instance without duplicates. Models enumeration is related to #SAT, the problem of computing the number of models for a given propositional formula. Model counting is the canonical #P-complete problem. On the practical side, for model counting, *SampleCount* a sampling based approach proposed by Gomes et al in [9], provides very good lower bounds with high confidence. Similarly, an efficient model enumeration algorithm has been proposed in [12, 5].

2 Motivating Example

To motivate our proposed framework, let us consider a simple example of a car dealer selling different cars bands with several possible options. For each car brand, several colors and types of fuels are available. The car dealer collected the preferences of four customers through a survey questionnaire. The first customer does not want red cars. The second wants a car with a diesel fuel, while the third wants a red car with gasoline fuel. Finally, the fourth customer prefers brand Peugeot cars. In addition to these customer desires, we also consider mutual exclusion constraints (mutex), allowing to express that each car must have only one color, one type of fuel and one car brand.

To express the different customer desires in propositional logic, we consider the following propositional variables: r (resp. b) represents red (resp. black) colors, p (resp. c) represents the Peugeot (resp. Citroen) car brand and d (resp. g) represents cars with diesel (resp. gasoline) fuel.

The mutex constraints are expressed by the following formula: $\mu = [(r \wedge \neg b) \vee (b \wedge \neg r)] \wedge [(g \wedge \neg d) \vee (d \wedge \neg g)] \wedge [(p \wedge \neg c) \vee (c \wedge \neg p)]$.

In Figure 1 (left hand side), for each customer c_i, we associate a propositional formula ϕ_{c_i} expressing its desires. We also provide the set of models satisfying both the desires of the customer and the mutex constraints ($\mathcal{M}(\phi_{c_i} \wedge \mu)$). The presentation of the models follows the variables ordering: $r \prec b \prec d \prec g \prec c \prec p$. In Figure 1 (right hand side), we give a graphical representation of the preferences of the four customers. This illustrative example highlights the expressiveness of

Customers	c_1	c_2	c_3	c_4
ϕ_{c_i}	$\neg r$	d	$g \wedge r$	p
$\mathcal{M}(\phi_{c_i} \wedge \mu)$	010101	011001		010101
	010110	011010		011001
	011001	101001	100101	100101
	011010	101010	100110	101001

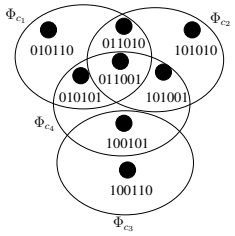


Fig. 1. Logical and Graphical Representation of Customers Preferences

logic-based data representation while allowing the possibility to define both user and background constraints.

3 Adapting Standard Clustering Algorithms

In this section, we present our extension of the well-known k-means and agglomerative hierarchical clustering algorithms to handle objects expressed as propositional formulas. Let us first fix some necessary notations and definitions.

We use $\mathcal{P}(k, \Phi)$ to denote the problem of clustering the set of propositional formulas $\Phi = \{\phi_1, \ldots, \phi_n\}$ into a set of k clusters with $k \leqslant n$. Let $\mathcal{C}$ be a family of sets over Φ. $\mathcal{C}$ is a solution of $\mathcal{P}(k, \Phi)$ if and only if $|\mathcal{C}| = k$, $\bigcup_{\mathcal{C}_i \in \mathcal{C}} \mathcal{C}_i = \Phi$ with $C_i \cap C_j = \emptyset$ for $1 \leqslant i < j \leqslant k$, and $\mathcal{M}(\bigwedge_{\phi \in \mathcal{C}_i} \phi) \neq \emptyset$ for every $\mathcal{C}_i \in \mathcal{C}$. We say that a clustering problem $\mathcal{P}(k, \Phi)$ is *consistent* if it admits a solution.

3.1 *k*-Means Algorithm for propositional formulas Clustering

Given a set of n data points in d-dimensional space $\mathbb{R}^d$ and a positive integer k, the k-means algorithm determines a set of k points in $\mathbb{R}^d$, called centers, so as to minimize an objective function such as the mean squared distance from each data point to its nearest center. To extend the k-means algorithm to clustering of objects described by propositional formulas, we need to define,

1. a distance between two formulas;
2. a centroid representing a given cluster;
3. an objective function to optimize.

Let us recall that a propositional formula ϕ can be equivalently expressed by its set of models $\mathcal{M}(\phi)$. With this representation in mind, one can consider that two formula ϕ_1 and ϕ_2 are similar if their set of common models $\mathcal{M}(\phi_1) \cap \mathcal{M}(\phi_2)$ is higher with respect to the remaining (distinctive) models $\mathcal{M}(\phi_1) \setminus \mathcal{M}(\phi_2) \cup \mathcal{M}(\phi_2) \setminus \mathcal{M}(\phi_1)$. This kind of similarity is related to the well-known contrast model of similarity proposed in a seminal paper by Tversky [15].

Definition 1 (Tversky [15]). *Let a and b be two objects described by two sets of features A and B respectively. Similarity between a and b, denoted $s(a, b)$, is defined as:*

$$s\,(a, b) = \frac{f\,(A \cap B)}{f\,(A \cap B) + \alpha f\,(A - B) + \beta f\,(B - A)} \quad \alpha, \beta \geqslant 0$$

The positive coefficients α and β reflects the weights given to the distinctive features of the two objects a and b. We usually assume that f is a matching function satisfying the additivity property $f(A \cup B) = f(A) + f(B)$, whenever A and B are disjoint. The ratio model defines a normalized value of similarity such that $0 \leqslant s(a, b) \leqslant 1$.

Contrast similarity model is particularly suitable in our context. To extend Definition 1, we consider the relationship between set operations and logical connectives. Indeed, the set union (resp. intersection) corresponds to disjunction (resp. conjunction). The difference between sets can be expressed using both conjunction and negation connectives, while the symmetric difference between sets can be expressed using the xor ($\oplus$) logical connective. Indeed, we have $\mathcal{M}(\phi_1) \setminus \mathcal{M}(\phi_2) \cup \mathcal{M}(\phi_2) \setminus \mathcal{M}(\phi_1) = \mathcal{M}((\phi_1 \wedge \neg\phi_2) \vee (\phi_2 \wedge \neg\phi_1)) = \mathcal{M}(\phi_1 \oplus \phi_2)$.

Using these relationships, we derive the following extension of the ratio model[16].

Definition 2. *Let a and b be two objects described by two propositional formulas ϕ_1 and ϕ_2 respectively. Similarity between a and b is defined as:*

$$s(a,b) = \frac{f(\phi_1 \wedge \phi_2)}{f(\phi_1 \wedge \phi_2) + \alpha f(\phi_1 \wedge \neg\phi_2) + \beta f(\phi_2 \wedge \neg\phi_1)} \quad \alpha, \beta \geqslant 0$$

In our context, as no distinction is made between the measure of $\phi_1 \wedge \neg\phi_2$ and $\phi_2 \wedge \neg\phi_1$, we derive the following similarity measure.

Definition 3. *Let a and b be two objects described by two propositional formulas ϕ_1 and ϕ_2 respectively. Similarity between a and b is defined as:*

$$s(a,b) = \frac{f(\phi_1 \wedge \phi_2)}{f(\phi_1 \wedge \phi_2) + \gamma f(\phi_1 \oplus \phi_2)}, \gamma \geqslant 0$$

From Definition 2 (resp. Definition 3), instantiating $\alpha = \beta = 1$ (resp. $\gamma = 1$), we derive a logic-based variant of the well known Jaccard similarity coefficient (resp. distance) [13]:

Definition 4. *Let a and b be two objects described by two propositional formulas ϕ_1 and ϕ_2 respectively. Similarity and distance between a and b or between ϕ_1 and ϕ_2 are defined respectively as:*

$$s_J(a,b) = s_J(\phi_1,\phi_2) = \frac{f(\phi_1 \wedge \phi_2)}{f(\phi_1 \vee \phi_2)} \text{ and } d_J(a,b) = 1 - s_J(a,b) = d_J(\phi_1,\phi_2)$$

As mentioned previously, considering the model based representation of propositional formulas, we define the function f as:

$$f : \left| \begin{array}{l} F_{\mathcal{P}} \longrightarrow \mathbf{N} \\ \phi \longmapsto |\mathcal{M}(\phi)| \end{array} \right.$$

Clearly, the function f satisfies the additive property. Indeed, we have $\mathcal{M}(\phi_1 \vee \phi_2) = \mathcal{M}(\phi_1) \cup \mathcal{M}(\phi_2)$. Computing f involves solving a #P-Complete model counting problem as discussed in Section 1.1.

Let us now define the representative of a cluster of propositional formulas.

Definition 5. *Let $\mathcal{C}_i$ be a cluster involving n_i formulas $\{\phi_{1_i}, \phi_{2_i}, \ldots, \phi_{n_i}\}$. We define the cluster representative (also called centroid) $\mathcal{O}_{\mathcal{C}_i}$ of the cluster $\mathcal{C}_i$ as:*

$$\mathcal{O}_{\mathcal{C}_i} = \phi_{1_i} \wedge \phi_{2_i} \wedge \ldots \wedge \phi_{n_i}$$

It is important to note that in our proposed extension, the goal is to group formulas into consistent clusters. Consequently, the *formula representing a given cluster must be consistent.*

We use the classical k-means objective function introduced in Definition 6

Definition 6. *Let $\mathcal{P}(k, \Phi)$ be the problem of clustering a set of propositional formulas $\Phi = \{\phi_1, \ldots, \phi_n\}$ to $k(\leqslant n)$ clusters $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_k\}$. The objective function is defined using Absolute-Error Criterion (AEC):*

$$C^* = \arg\min_{\mathcal{C}} \sum_{i=1}^{k} \sum_{\phi \in \mathcal{C}_i} d_J(\phi, \mathcal{O}_{\mathcal{C}_i}) \tag{1}$$

Our clustering algorithm of a set of propositional formulas can now be derived from the classical k-means algorithm using the new components (distance, centroid and objective function) defined above.

3.2 Hierarchical Agglomerative Algorithm for propositional formulas Clustering

Hierarchical algorithms can behave better than the k-means. The base idea of hierarchical agglomerative algorithms is to build a dendrogram such that at each level the two closest clusters are merged. By applying a hierarchical algorithm, we will ensure that if there are two objects that are closest to each other, they will necessarily be in the same cluster. In this adaptation, the similarity between two clusters is identical to the similarity between their representatives. Similarly to Definition 5, the conjunction of all formulas in a cluster represents its centroid. To merge clusters, we combine the two clusters with the smallest centroid distance. Using this adaptation, we can applay a standard hierarchical agglomerative algorithm on data represented as boolean formulas as illustrated in figure 2.Note that this algorithm needs at least $\mathcal{O}(n^2)$ calls to a # SAT oracle.

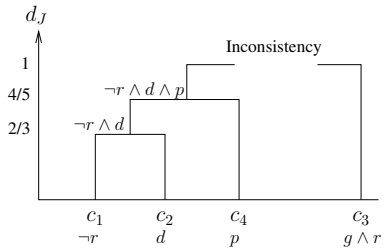


Fig. 2. Agglomerative Clustering on the Car Dealer Example

4 Divisive Algorithm for Model Based Representation

As mentioned previously, when we consider the problem of clustering a set of formulas $\Phi = \{\phi_1, \phi_1, \ldots, \phi_n\}$ without common model, i.e., $\Phi \vdash \bot$, agglomerative algorithm and k-means can fail to find a clustering with the desired number of clusters. In the sequel, we propose a top-down hierarchical (or divisive) algorithm for clustering a set of propositional formulas. Our proposed adaptation makes use of the well-known minimum hitting sets problem, that we recall.

Definition 7. *H is a hitting set of a set of sets Ω if $\forall S \in \Omega$, $H \cap S \neq \emptyset$. A hitting set H is irreducible if there is no other hitting set H' s.t $H' \subset H$. H is called minimum hitting set if there is no hitting set H' such that $|H'| < |H|$.*

Example 1. Let $\Phi = \{\phi_1, \phi_2, \phi_3\}$ be a set of propositional formulas such that $\mathcal{M}(\phi_1) = \{m_1, m_2, m_3\}$, $\mathcal{M}(\phi_2) = \{m_1, m_4\}$ and $\mathcal{M}(\phi_3) = \{m_3, m_5\}$. The set $H = \{m_1, m_3\}$ is a minimum and irreducible hitting set of the models of Φ.

In our adaptation, we choose the worst cluster to divide according to the following quality measure.

Definition 8. *Let $\mathcal{C}_i = \{\phi_{1_i}, \ldots, \phi_{n_i}\}$ be a cluster of n_i propositional formulas. We define the quality of $\mathcal{C}_i$ as:*

$$\mathcal{Q}(\mathcal{C}_i) = \frac{|\mathcal{M}(\phi_{1_i} \wedge \cdots \wedge \phi_{n_i})|}{|\mathcal{M}(\phi_{1_i} \vee \cdots \vee \phi_{n_i})|}$$

The quality of a cluster is obtained by extending the similarity measure between two formulas to a set of formulas. Indeed, a cluster is qualified to be of poor quality, when its formulas admits a great number of models while sharing a small number of models. Consequently, the worst cluster is obtained as follows:

$$\mathcal{C}_i^* = \underset{\mathcal{C}_i \in \mathcal{C}}{\operatorname{argmin}}\ \mathcal{Q}(\mathcal{C}_i)$$

Definition 9. *Let Φ be a set of propositional formulas and $\mathcal{I}$ a Boolean interpretation. We define the subset of formulas of Φ sharing the model $\mathcal{I}$ as* $\mathcal{S}(\mathcal{I}, \Phi) = \{\phi \in \Phi \mid \mathcal{I} \vDash \phi\}$

To build consistent clusters, Algorithm 1 starts by computing a minimum hitting set H of the set of sets of models of the formulas in Φ (line 1). The main idea behind our algorithm is to use the models of the computed minimum hitting set to divide a cluster into several consistent clusters. Each cluster is obtained by selecting for each model m of the minimum hitting set, the set of formulas admitting m as a model. In this way, the formulas in the obtained clusters share at least one model. If the size of the minimum hitting set H is greater than k, then no clustering is possible, and the algorithm returns an empty set (line 3), otherwise a consistent clustering can be obtained. In this last case, the algorithm starts by a clustering $\mathcal{C}$ where all the formulas in Φ are grouped into a single cluster (line 6). We start an iterative top-down divisive process (lines 7-20), until generating k clusters. At each iteration, we choose a cluster to divide (line 8) which is one of those with the worst quality (see Definition 8). Then, we build Ω the set of sets of models of the formulas involved in the selected cluster, while removing the set of common models M (lines 9-10). A minimum hitting set H of Ω is then computed (line 11). It is important to note that by removing the common models M from the models of each formula of the selected cluster, we avoid the trivial minimum hitting sets of size 1. Now, we use the hitting set H to divide the chosen cluster $\mathcal{C}_i^*$ into $|H|$ clusters (line 12). Indeed, for each model m in H, we associate a cluster Ψ_m made of formulas of $\mathcal{C}_i^*$ sharing the model m. In this way, we maintain the consistency property on each new cluster Ψ_m. Now, we substitute in $\mathcal{C}$ the cluster of poor quality $\mathcal{C}_i^*$ with the new set of clusters (line 18). However, this is only done when the size of the new

Algorithm 1: Model-Based Divisive Algorithm for Clustering Boolean Formulas

Input: A set of formulas $\Phi = \{\phi_1, \ldots, \phi_n\}$ and an integer $k \geqslant 1$
Output: A set of clusters $\mathcal{C} = \{C_1, \ldots, C_k\}$

1 $H \leftarrow minHittingSet(\{\mathcal{M}(\phi_1), \ldots, \mathcal{M}(\phi_n)\})$;
2 **if** $(|H| > k)$ **then**
3 | **return** $\emptyset$;
4 **end**
5 **else**
6 | $\mathcal{C} \leftarrow \{\Phi\}$;
7 | **while** $(|\mathcal{C}| \; != k)$ **do**
8 | | $\mathcal{C}_i^* = \{\phi_{i_1} \ldots \phi_{i_{n_i}}\} \leftarrow \underset{\mathcal{C}_i \in \mathcal{C}, |\mathcal{C}_i|>1}{\arg\min} \mathcal{Q}(\mathcal{C}_i)$, $\triangleright \; n_i = |\mathcal{C}_i^*|$;
9 | | $M = \mathcal{M}(\phi_{i_1}) \cap \cdots \cap \mathcal{M}(\phi_{i_{n_i}})$;
10 | | $\Omega = \{\mathcal{M}(\phi_{i_1}) \setminus M, \ldots, \mathcal{M}(\phi_{i_{n_i}}) \setminus M\}$;
11 | | $H \leftarrow minHittingSet(\Omega)$;
12 | | $\forall m \in H, \Psi_m \leftarrow \mathcal{S}(m, \mathcal{C}_i^*)$;
13 | | **if** $(|\mathcal{C}| + |H| - 1 > k)$ **then**
14 | | | $\Psi \leftarrow merge(\{\Psi_{m_1}, \ldots, \Psi_{m_{|\mathcal{C}|+|H|-1-k}}\})$;
15 | | | $\mathcal{C} \leftarrow (\mathcal{C} \setminus \mathcal{C}_i^*) \cup \{\Psi\} \cup \{\Psi_{m_{|\mathcal{C}|+|H|-k}}, \ldots, \Psi_{m_{|H|}}\}$
16 | | **end**
17 | | **else**
18 | | | $\mathcal{C} \leftarrow (\mathcal{C} \setminus \mathcal{C}_i^*) \cup \{\Psi_{m_1}, \ldots, \Psi_{m_{|H|}}\}$
19 | | **end**
20 | **end**
21 **end**
22 $\mathcal{C} \leftarrow eliminateOverlap(\mathcal{C})$;
23 **return** $\mathcal{C}$

clustering does not exceed k (line 13); otherwise to obtain exactly k clusters, we merge (function `merge`) the first $|\mathcal{C}| + |H| - (k+1)$ of these new clusters (line 14) before applying substitution (line 15). Note that in the divisive step (line 12), a formula can belong to several new clusters. The reason comes from the fact that a given formula can share several models of the minimum hitting set. Consequently, a last step is then performed to produce non overlapping clusters (line 20 - function *eliminateOverlap*). To do this, for each formula occurring in several clusters, we keep it in the cluster with the best quality, while removing it in the remaining clusters. Obviously, depending on applications, overlapping clusters might be more suitable. In this case, one only need to skip the call to the overlap elimination function.

Algorithm 1, involves $\mathcal{O}(n)$ calls to model enumeration problem (line 1), $\mathcal{O}(k)$ calls to # SAT oracle (line 8) and $\mathcal{O}(k)$ calls to minimum hitting set problem (line 1 and 11).

Let us now gives some interesting properties of our propositional formulas based divisive algorithm. The first one states the correctness of our algorithm.

Proposition 1. *If $\mathcal{P}(k, \Phi)$ is consistent, then Algorithm 1 produces a clustering.*

The proof trivially follows from the previous detailed explanation on how the algorithm operates.

The second property allows us to establish that two equivalent formulas might be located in the same cluster when overlaps between clusters are allowed.

Proposition 2. *Let $\mathcal{P}(k, \Phi)$ be a clustering problem with overlaps, $\mathcal{C}$ a clustering of $\mathcal{P}(k, \Phi)$ and $\phi_1, \phi_2 \in \Phi$. If $\phi_1 \equiv \phi_2$ then $\forall \mathcal{C}_i \in \mathcal{C}$, $\phi_1 \in \mathcal{C}_i$ iff $\phi_2 \in \mathcal{C}_i$.*

The last property generalizes the previous property to the case of two formulas where one is a logical consequence of the other.

Proposition 3. *Let $\mathcal{P}(k, \Phi)$ be a clustering problem with overlaps, $\mathcal{C}$ a clustering of $\mathcal{P}(k, \Phi)$ and $\phi_1, \phi_2 \in \Phi$. If $\phi_1 \vdash \phi_2$ then $\forall \mathcal{C}_i \in \mathcal{C}$, if $\phi_1 \in \mathcal{C}_i$ then $\phi_2 \in \mathcal{C}_i$.*

5 SAT encoding for a Bounded Consistent Clustering

As discussed in the previous section, when the propositional formulas are not represented by their models, our proposed model based divisive algorithm requires $\mathcal{O}(n)$ calls to model enumeration oracle, to compute the set of models of each formula. Such set of models might be of exponential size in the worst case. In addition to these limitations, one also need to compute a minimum hitting set of a set of sets of models ($\mathcal{O}(k)$ calls). In this section, we present an alternative approach that significantly reduces the overall complexity of our Algorithm. To this end, we introduce a SAT-based encoding that allows to find a bounded consistent clustering of a given set of propositional formulas.

Let $\Phi = \{\phi_1, \ldots, \phi_n\}$ be a set of propositional formulas and k a positive integer. To define our encoding, we associate to each propositional variable p appearing in Φ a set of k fresh propositional variables, denoted $p^1, \ldots, p^k$. Then, for every formula $\phi_i \in \Phi$ and $j \in \{1, \ldots, k\}$, we use ϕ_i^j to denote the formula obtained from ϕ_i by replacing each propositional variable p with the fresh variable p^j. The formula ϕ_i^j is used to model the fact that ϕ_i is in the j^{th} cluster.

The following formula expresses that each formula in Φ has to be true in at least one consistent cluster:

$$\bigwedge_{i=1}^{n} \left(\bigvee_{j=1}^{k} \phi_i^j \right) \tag{2}$$

One can easily see that (2) is satisfiable if and only if Φ can be partitioned in k consistent clusters. It is worth noting that in a model of (2) a formula can belong to more than one cluster. To obtain a bounded consistent clustering from a model m, we only have to consider for each formula $\phi_i \in \Phi$ a single positive integer j in the set $\{1 \leqslant j \leqslant k \mid m(\phi_i^j) = 1\}$. This problem can be avoided by reformulation. To this end, we associate to each formula ϕ_i in Φ a set of k fresh propositional variables, denoted $q_{\phi_i}^1, \ldots, q_{\phi_i}^k$. The variable $q_{\phi_i}^j$ is used to represent the fact that ϕ_i is in the j^{th} cluster by using the following formula:

$$\bigwedge_{i=1}^{n} \left(\bigwedge_{j=1}^{k} q_{\phi_i}^j \Leftrightarrow \phi_i^j \right) \tag{3}$$

Then, to express that each formula in Φ belongs to exactly one consistent cluster, we use the following formula:

$$\bigwedge_{i=1}^{n}(\sum_{j=1}^{k} q_{\phi_i}^{j} = 1) \tag{4}$$

Our second SAT encoding of the bounded consistent clustering problem $\mathcal{P}(k, \Phi)$ is defined by the formula $\mathcal{P}_{SAT}(k, \Phi) = (3) \wedge (4)$. From a model m of $\mathcal{P}_{SAT}(k, \Phi)$, a clustering can be easily extracted. Indeed, if $m(q_{\phi_i}^{j}) = true$ then $\phi_i \in \mathcal{C}_j$ otherwise $\phi_i \notin \mathcal{C}_j$.

Definition 10. *Let $\Phi = \{\phi_1, \ldots, \phi_n\}$. C is called a minimum consistent clustering of Φ if there is no consistent clustering $\mathcal{C}'$ of Φ such that $|\mathcal{C}'| < |\mathcal{C}|$.*

As we can observe, clustering propositional formulas can be done using Algorithm 1 by replacing the computation of the minimum hitting set with the computation of the minimum consistent clustering (Definition 10) using $\mathcal{P}_{SAT}(k, \Phi)$. Similarly to Algorithm 1, Properties 1, 2 and 3 holds.

6 Experimentation

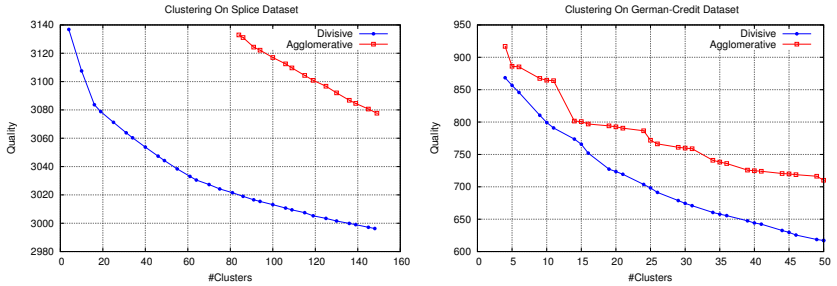
In this section, we carried out an experimental evaluation of the performance of our divisive and agglomerative algorithms for the clustering of a set of propositional formulas. Our goal is to assess the feasibility and effectiveness of our proposed framework.

We performed our experiments on a machine with Intel Core2 Quad CPU of 2.66GHz and 8G of RAM. Our first aim is to compare the performance of our divisive and agglomerative algorithms. To this end, We consider two datasets `splice`, and `german-credit`[1]. We consider each data set as a set of transactions, where each transaction is a formula (a set of models). Consequently, an item is assimilated to a model.

Figure 3 shows the performances of agglomerative (Algorithm **??**) and divisive (Algorithm 1) methods on the problem of clustering transaction databases. First, our divisive algorithm outperforms the agglomerative algorithm on `splice` and `german-credit`. Nevertheless, as illustrated in section 3.2, the agglomerative algorithm is unable to find a clustering all the time. This is the case on `splice` data, where such approach can not provide clustering answer when the number of desired clusters is less than 84.

To further investigate the expressiveness and the ability of our approach to scale, we enlarge our experiments of the previous problem by studying the clustering of a set of formulas resulting from a random-generated poll with 100 to 1000 participants where each participant is invited to report its preferences. The questions of the poll are organized in four levels. At the first level, the participant is invited to select its 3 preferred options among 5. According to the

[1] https://dtai.cs.kuleuven.be/CP4IM/



splice german-credit

Fig. 3. Model approach: Agglomerative vs Divisive

preferences of the participant, she/he is invited to select other preferences from the second level and so on until the last level (level 4). For illustration, assume that in the first level we consider a set S of courses (e.g. Artificial Intelligence, Data Mining, Databases, Networks and Web Programming). A student selects three courses from S (level 1). Then, for each selected course, she/he chooses chapters (level 2), and so on. The preferences of each participant are encoded as a propositional formula (the resulting formulas have between 567 and 1813 models). Agglomerative approach is not considered since it can not guaranty to find a clustering solution if it exists.

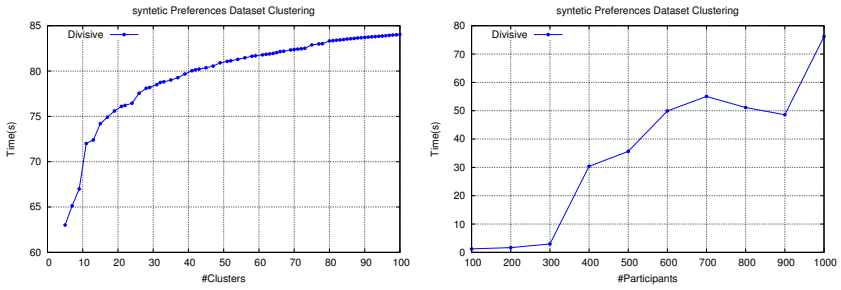


Fig. 4. Time vs #Clusters vs #Participants

The time needed to obtain a clustering, Figure 4, does not exceed 100 seconds for all values of k. This shows that our approach scale well. Finally, we study the evolution of the time needed to find a clustering when the number of clusters is fixed to 20 and the number of participants is varied from 100 to 1000 (Figure 4). Here again the time needed is reasonable, i.e., less than 100 seconds.

7 Conclusion et perspectives

In this work we introduced the concept of consistent clustering propositional formulas. We show how well-known k-means, agglomerative and divisive algorithms

can be adapted to this new framework. We then, propose two new solutions. The first one called model based, assume that the set of models of each formula are given. We then show how the hitting set notion is used to efficiently give a consistent clustering. In the second part, we propose an encoding into SAT of the divisive algorithm that make a linear number of calls to a #SAT oracle to count the set of models during the clustering steps. As a future work, we plan to explore other similarity measure, to define intuitive distance between propositional formulas. Improving our divisive algorithm by exploiting efficiently the overlaps deserves further investigation.

References

1. C. C. Aggarwal and C. K. Reddy. *Data clustering: algorithms and applications*. CRC Press, 2013.
2. P. Berkhin. A survey of clustering data mining techniques. In J. Kogan, C. K. Nicholas, and M. Teboulle, editors, *Grouping Multidimensional Data - Recent Advances in Clustering*, pages 25–71. Springer, 2006.
3. L. Billard and E. Diday. *Symbolic Data Analysis: Conceptual Statistics and Data Mining*. John Wiley & Sons, May 2012.
4. H. H. Bock. *Analysis of Symbolic Data: Exploratory Methods for Extracting Statistical Information from Complex Data*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2000.
5. S. Chakraborty, K. Meel, and M. Vardi. A scalable approximate model counter. In *CP'2013)*, pages 200–216, 2013.
6. F. d. A. de Carvalho, M. Csernel, and Y. Lechevallier. Clustering constrained symbolic data. *Pattern Recognition Letters*, 30(11):1037–1045, 2009.
7. R. M. de Souza and F. d. A. De Carvalho. Clustering of interval data based on city–block distances. *Pattern Recognition Letters*, 25(3):353–365, 2004.
8. E. Diday and F. Esposito. An introduction to symbolic data analysis and the SODAS software. *Intell. Data Anal.*, 7(6):583–601, 2003.
9. C. P. Gomes, J. Hoffmann, A. Sabharwal, and B. Selman. From sampling to model counting. In *IJCAI'1997*, pages 2293–2299, 2007.
10. K. C. Gowda and E. Diday. *New Approaches in Classification and Data Analysis*, chapter Symbolic Clustering Algorithms using Similarity and Dissimilarity Measures, pages 414–422. Springer Berlin Heidelberg, Berlin, Heidelberg, 1994.
11. L. Hotz, A. Felfernig, M. Stumptner, A. Ryabokon, C. Bagley, and K. Wolter. Chapter 6 - configuration knowledge representation and reasoning. In *Knowledge-Based Configuration*, pages 41 – 72. Morgan Kaufmann, 2014.
12. S. Jabbour, J. Lonlac, L. Sais, and Y. Salhi. Extending modern SAT solvers for models enumeration. In *IEEE-IRI'2014*, pages 803–810, 2014.
13. P. Jaccard. The distribution of the flora of the alpine zon. *New Phytologist*, 11:37–50, 1912.
14. R. S. Michalski. Knowledge acquisition through conceptual clustering: A theoretical framework and an algorithm for partitioning data into conjunctive concepts. *Journal of Policy Analysis and Information Systems*, 4(3):219–244, 1980.
15. A. Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1977.
16. A. Tversky. *Preference, Belief, and Similarity*. The MIT Press, November 2003.