ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240672

On the Discovery of Conceptual Clustering Models Through Pattern Mining

Motaz Ben Hassine [1,2,*], **Saïd Jabbour** [1], **Mourad Kmimech** [3], **Badran Raddaoui** [4] **and Mohamed Graiet** [5]

[1]CRIL, University of Artois & CNRS, Lens, France
[2]University of Monastir, UR-OASIS-ENIT, Monastir, Tunisia
[3]EFREI, Paris-Panthéon-Assas University, France
[4]SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, France
[5]LS2N Nantes, Nantes, France

Abstract. Conceptual clustering is a well-studied research area in the field of unsupervised machine learning. It aims to identify disjoint clusters, where each cluster represents a collection of similar transactions described by a common pattern. The first phase of earlier conceptual clustering methods relies on the enumeration of closed patterns. Nevertheless, the extraction of such patterns can be challenging, primarily due to their rigorous nature. Indeed, closed patterns can be not frequent or fail to cover all the transactions within a cluster. To overcome this issue, this paper presents a novel approach based on the relaxation of frequent patterns called *k-relaxed frequent patterns*. Then, we introduce a propositional satisfiability method for enumerating such patterns. Afterwards, we employ an integer linear programming approach to compute the set of disjoint clusters. Finally, we demonstrate the efficiency of our approach through an extensive experiments conducted on several popular real-life datasets.

1 Introduction

Data clustering is a popular machine learning task that helps to gain a deeper understanding of the data. It plays a significant role in numerous practical applications, including document and text classification, customer segmentation, image and video analysis, genomic data analysis, recommender systems, and community detection in social networks, among others. In the literature, numerous proposals have been studied for data clustering. The majority of existing approaches fall under the unsupervised machine learning techniques and can be broadly categorized into *hierarchical* and *conceptual* clustering methods. The first category can be classified into agglomerative or divisive approaches [29]. Agglomerative clustering techniques, e.g. [2, 41, 18], are iterative methods for clustering data points. At each step, new data is grouped based on a (dis)similarity function and an optimization metric. On the other hand, divisive methods involve dividing data points into clusters based on a given (dis)similarity function. The process repeats until a stopping criterion is met, guided by an optimization function, similar to the k-Means approach [28] or also the overlapping k-Means (i.e. Neo-k-Means) [42]. Note that several other hierarchical clustering approaches have been introduced so far (e.g. [38, 13, 1, 43, 31]). Moreover, other methods, not falling within the classes of divisive, agglomerative, or conceptual clustering approaches, have been proposed, such as the Spectral Clustering method [39] and the BIRCH approach [44]. The second category consists of *conceptual clustering* approaches, with the latter being of particular interest in this paper. Conceptual clustering aims to group data points, each described with certain features, according to their descriptions into the same cluster. More precisely, given a dataset $\mathcal{D}$ (with m transactions and f features) and let β be the desired number of clusters, the goal of conceptual clustering is to identify β disjoint clusters that collectively encompass all the data in $\mathcal{D}$ while corresponding to meaningful concepts. Numerous approaches have been proposed to address the problem of conceptual clustering, e.g. [33, 14, 15]. Some of these existing techniques are based on declarative approaches using boolean satisfiability problem (SAT) [32], constraint programming (CP) [16, 30] or also on Integer Linear Programming (ILP) [36, 37, 7]. These earlier approaches employed data mining techniques. Indeed, the initial phase of these methods, such as [36], relies on enumerating all classical closed patterns. However, in many cases, the extraction process of closed patterns can be challenging due to their rigorous nature. Indeed, they may not appear frequently in the dataset or fail to cover all the transactions within a cluster. To overcome this issue, we propose in this paper relaxing these patterns in order to enhance the conceptual clustering task. The main technical contributions of the paper are three-folds:

1. We present the k-Relaxed Frequent Pattern (or k-RFP in short) model, and we use a SAT based encoding to enumerate these patterns.
2. Leveraging ILP, we select the best clusters that match those patterns based on an objective function.
3. We show the efficiency of the proposed clustering approach through an extensive experiments conducted on various popular real-life datasets.

The paper is organized as follows: we introduce a brief overview on propositional logic, the propositional satisfiability problem, and the conceptual clustering problem in Section 2. Afterwards, Section 3 presents our motivation and discusses the k-RFP and the ILP models for the conceptual clustering problem. Section 4 provides our experimental study on real-life datasets. Finally, Section 5 concludes the paper with hints on forthcoming work.

* Corresponding Author. Email: benhassine@cril.fr.

2 Preliminaries

This section introduces a brief overview on propositional logic, the propositional satisfiability and the conceptual clustering problems.

2.1 Propositional Logic & SAT

We consider a propositional language, denoted as $\mathcal{L}$, constructed inductively from a countable set $\mathcal{PS}$ comprising propositional letters, the Boolean constants $\top$ (representing *true* or 1) and $\bot$ (representing *false* or 0), and the well-known logical connectives $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$ in the usual manner. The symbols x, y, z, etc., iterate over the elements of $\mathcal{PS}$. Propositional formulas of $\mathcal{L}$ are represented by Φ, Ψ, Γ, etc. A literal is either a propositional variable (x) of $\mathcal{PS}$ or its negation ($\neg x$). A clause is a (finite) disjunction of literals, while a term is a (finite) conjunction of literals. A clause which contains only one literal is referred to as a unit clause. For a formula Φ from $\mathcal{L}$, let $\mathcal{Z}(\Phi)$ indicates the symbols of $\mathcal{PS}$ appearing in Φ. A conjunctive normal form (CNF) formula is a (finite) conjunction of clauses, and a formula in disjunctive normal form (DNF) is a (finite) disjunction of terms. A Boolean interpretation $\mathcal{I}$ of a propositional formula Φ is a mapping from $\mathcal{Z}(\Phi)$ to $\{0, 1\}$. If $\mathcal{I}(\Phi) = 1$, then $\mathcal{I}$ is referred to as a model of Φ.

The propositional satisfiability problem (**SAT** in short) involves determining whether a CNF formula has a model. Acknowledged as an NP-Complete problem, SAT has demonstrated successful applications in various practical domains, such as itemset mining [8, 17, 9, 23, 19], association rule mining [21, 27], conceptual clustering [32], overlapping community detection in networks [24, 25, 26], and others.

2.2 Conceptual Clustering Problem

This subsection introduces the basics of the problem of conceptual clustering. We start with some preliminary information on pattern mining before presenting the concept of conceptual clustering.

Let Ω be a universe of items (or symbols) that may represent articles in a supermarket, web pages, or a collection of attributes or events. The letters a, b, c, etc., will be employed to iterate over the elements of Ω. A **classical pattern** (or simply a pattern or an itemset) is a subset of items in Ω, that is, $P \subseteq \Omega$ with $P \neq \emptyset$. The set of all patterns over Ω, denoted as 2^{Ω}, are represented by the capital letters P, Q, R, etc. A **dataset** is a finite non-empty set of transactions or records $\mathcal{D} = \{T_1, T_2, \ldots, T_m\}$. Given a transaction database $\mathcal{D}$ and a pattern P, the **cover** of P in $\mathcal{D}$ is a mapping $2^{\Omega} \mapsto 2^{\mathcal{D}}$ which maps each pattern P to a set of transactions in $\mathcal{D}$ containing P. More formally, $\mathsf{C}(P, \mathcal{D}) = \{i \in [1..m] \mid T_i \in \mathcal{D} \text{ and } P \subseteq T_i\}$. The cardinality of the cover of a pattern P represents its **support** (also called *frequency*). We write $\mathsf{Supp}(P, \mathcal{D})$ for the support of P in the dataset $\mathcal{D}$, i.e., $\mathsf{Supp}(P, \mathcal{D}) = |\mathsf{C}(P, \mathcal{D})|$. Given a pattern P and a dataset $\mathcal{D}$, P is called a **closed pattern** iff $\nexists\ R$ s.t. $P \subset R$ and $\mathsf{Supp}(R, \mathcal{D}) = \mathsf{Supp}(P, \mathcal{D})$. A **Generalized Disjunctive Pattern** (GDP) is a composite structure, represented as $[P_1, \ldots, P_p]$, that encapsulates a collection of patterns in a disjunctive fashion. This distinctive square bracket notation is intentionally chosen to distinguish a GDP from classical patterns [34]. In essence, a GDP $[P] = [P_1, \ldots, P_p]$ can be equivalently expressed as a formula in DNF: $\bigvee_{1 \leq i \leq p} (\wedge_{a \in P_i} a)$.

Definition 1. *The support of a GDP in the dataset $\mathcal{D}$ is defined by the following equation:*

$$\mathsf{Supp}([P], \mathcal{D}) = \frac{|\bigcup_{P_i \in P} \mathsf{C}(P_i, \mathcal{D})|}{|\mathcal{D}|}$$

Next, let us introduce the notion of a *concept* as follows:

Definition 2. *Given a dataset $\mathcal{D}$, a closed pattern P, and a subset of transactions $O \subseteq \mathcal{D}$ s.t. $|O| = r$. Then, the pair (O, P) is called a* **concept** *iff $\forall\ T_i \in O$ we have $P \subseteq T_i$ with $1 \leq i \leq r$.*

Now, we are ready to define the problem of *conceptual clustering*.

Definition 3. *Let $\mathcal{D}$ be a dataset, and β a positive integer where $\beta > 1$. The* **conceptual clustering problem** *aims to find β disjoint clusters $Cl = \{O_1, O_2, \ldots, O_\beta\}$ that cover $\mathcal{D}$ and corresponds to concepts.*

3 Conceptual Clustering Approach via k-RFP

In this section, we begin by presenting a motivating example to elucidate more concretely the issue of classical patterns in conceptual clustering. Following that, we discuss the process of extracting k-RFP from the input data using a SAT-based encoding. Then, we introduce our ILP models for optimization and acquiring the best clusters utilizing k-RFPs.

3.1 Motivating Example

Our work is motivated by the inherent rigidity of classical patterns in addressing the conceptual clustering problem. To be precise, in certain cases, classical patterns may not manifest frequently enough or fail to cover transactions to adequately characterize a high-quality clusters. To illustrate, let us consider the following data set depicted by Example 1.

Example 1. *Consider the dataset $\mathcal{D}$ outlined in Table 1. Assume that the target number of clusters is $\beta = 2$, with the desired clusters identified as $\{T_1, T_2, T_3, T_4\}$ and $\{T_5, T_6, T_7, T_8\}$. Our objective is to discover the patterns that thoroughly cover the two desired clusters. When attempting with classical patterns, let us consider the two patterns, $P = \{d\}$ and $R = \{e\}$, employed for cluster description. These patterns exhibit a support of 3 for both P and R, representing the closest coverage to generate clusters akin to the desired ones. Patterns P and R appear respectively in clusters $\{T_2, T_3, T_4\}$ and $\{T_6, T_7, T_8\}$. However, T_1 and T_5 remain uncovered. Importantly, any alternative pattern discovered would cover transactions less or equal than P and R. Then, it is impossible to achieve the desired cluster formation. This discrepancy poses a challenge. To overcome this, our goal is to use relaxed patterns.*

Table 1. A sample dataset $\mathcal{D}$

Transactions	Items							
T_1	a	b	c					
T_2	a	b		d				
T_3	a		c	d				
T_4		b	c	d				
T_5						f	g	h
T_6					e		g	h
T_7					e	f		h
T_8					e	f	g	

To address the challenges outlined in Example 1, we propose to leverage a novel pattern model called k-RFP. To enumerate such patterns, we employ a SAT-based encoding, previously used in the context of association rule mining [27].

3.2 k-Relaxed Frequent Pattern

It has been shown in [36] that classical patterns are essential for conceptual clustering. However, classical patterns can be not frequent or fail to cover transactions in a given input data.

To address this issue, we first revisit the traditional pattern model by recalling the concepts of k-cover and k-support [27].

Definition 4. *Let $\mathcal{D}$ be a dataset and k a positive integer. Then, the* **k-cover** *of a pattern P is $\mathsf{C}^k(P,\mathcal{D}) = \{i \in [1..m] \mid T_i \cap P \neq \emptyset$ and $|P \setminus T_i| \leq k\}$. The* **$k$-support** *of P is defined as usual as: $\mathsf{Supp}^k(P,\mathcal{D}) = |\mathsf{C}^k(P,\mathcal{D})|$.*

Unlike the traditional pattern model, which necessitates the complete inclusion of a pattern within a transaction, the k-cover relaxes this constraint. It allows transactions to have up to k missing items from the pattern to be considered as matching the pattern. The k-support of a pattern, in turn, tells us how many transactions in the dataset match these relaxed criteria of the k-cover.

Note that for a given pattern P, one can derive a GDP $[P_1, P_2, \ldots, P_p]$, where each P_i is defined as $[P \cap T_i \mid i \in \mathsf{C}^k(P,\mathcal{D})]$. Now, we can evaluate the significance of a pattern using the k-support concept.

Definition 5. *Let $\mathcal{D}$ be a dataset and α a support threshold s.t. $\alpha > 0$. Then, a pattern P is called a* **k-Relaxed Frequent Pattern** *(k-RFP) iff $\mathsf{Supp}^k(P,\mathcal{D}) \geq \alpha$.*

In informal terms, Definition 5 states that a pattern is considered frequent according to the k-support if it satisfies a specified minimum support threshold. Additionally, we characterize a pattern P as closed w.r.t. the k-support iff for all $P \subset R$, $\mathsf{Supp}^k(R,\mathcal{D}) < \mathsf{Supp}^k(P,\mathcal{D})$.

In this paper, our goal is to identify all closed k-RFPs that will be used for conceptual clustering. Specifically, given a dataset $\mathcal{D}$, a minimum support threshold α, and a positive integer k, we employ a SAT based encoding to discover the complete set of closed k-RFPs. Our approach involves translating the problem of extracting closed k-RFPs into the enumeration of all possible models of a corresponding CNF formula. Each model corresponds exactly to a closed k-RFP. It is important to note that separating the modeling and solving steps provides a flexible means to evolve the problem specification. This allows us to easily introduce new constraints to the symbolic encoding. Additionally, advancements in SAT solving technology contribute to the optimization of the solving step. However, it is important to acknowledge that the efficiency of the solving phase is significantly influenced by how the problem is encoded. The challenge lies in creating the most suitable encoding that balances efficiency and conciseness while ensuring correctness and completeness. This entails making judicious choices regarding propositional variables and logical constraints, as well as their reformulation into CNF.

Next, let us outline the SAT-based encoding to compute the (closed) k-RFPs from input data. First, we establish a clear connection between the models of the SAT encoding and the set of (closed) k-RFPs. This connection is made by associating each item $a \in \Omega$ and each transaction $T_i \in \mathcal{D}$ with respective propositional variables x_a and o_i. Second, we introduce a propositional formula consisting in a set of constraints allowing a one-to-one mapping between this formula and the set of k-RFPs.

Cover Constraint. The first This constraint is expressed as follows:

$$\bigwedge_{T_i \in \mathcal{D}} (o_i \leftrightarrow (\sum_{a \in \Omega \setminus T_i} x_a \leq k)) \tag{1}$$

Constraint (1) guarantees that a transaction T_i supports the k-RFP candidate when at most k items of k-RFP are not present in T_i.

Frequency Constraint. The second constraint is the *Frequency constraint* expressed as follows:

$$\sum_{i=1}^{m} o_i \geq m \times \alpha \tag{2}$$

Constraint (2) guarantees that a k-RFP covers at least $m \times \alpha$ transactions. Where α is a percentage.

Closure Constraint. The third constraint is the *Closure constraint* expressed as follows:

$$\bigwedge_{a \in \Omega} (\neg x_a \rightarrow \bigvee_{T_i \in \mathcal{D}, a \in T_i} (o_i \wedge \sum_{b \in \Omega \setminus T_i} x_b = k)) \tag{3}$$

Constraint (3) ensures that the item a cannot be part of the candidate k-RFP if its inclusion violates the cover constraint in at least one transaction.

Item Frequency Constraint. The fourth constraint is the *Item Frequency Constraint* expressed as follows:

$$\bigwedge_{T_i \in \mathcal{D}} (x_a \rightarrow (\sum_{T_i \in \mathcal{D} \mid a \in T_i} o_i \geq \gamma)) \tag{4}$$

Constraint (4) ensures that each item appears at least γ times in a pattern.

For Constraint (4), the objective is to minimize the number of identified k-RFPs and then extract only the most relevant k-RFPs.

It is important to emphasize that these constraints will be translated into CNF. Almost constraints involve cardinality expressions of the form $\sum_{i=1}^{n} y_i \geq \theta$, $\sum_{i=1}^{n} y_i \leq \theta$, or $\sum_{i=1}^{n} y_i = \theta$. Various encoding techniques have been proposed to translate cardinality constraints into CNF, see e.g., [40, 12, 3, 5].

This encoding consists of four constraints using cardinality constraints by using the sequential unary counter encoding [40, 22] having a complexity of $O(b \times (n-b))$, for a cardinality constraint of the form $x_1 + \ldots + x_n \geq b$. The complexity of each constraint (1), (2), (3) and (4) is respectively $O(k \times (n-k) \times m)$, $O(m^2 \times \alpha \times (1-\alpha))$, $O(n+m \times k \times (n-k))$ and $O(n \times \gamma(m-\gamma))$. Considering the worst-case complexity, then we have $O(n \times m^2 + m \times n^2)$ for $\alpha = 0.5$, $\gamma = \frac{m}{2}$, $k = \frac{n}{2}$.

Proposition 1. *The propositional formula $\Phi^{\alpha,\gamma,k} = (1) \wedge (2) \wedge (3) \wedge (4)$ encodes the constraints for closed k-RFPs extraction problem where each k-RFP covers at least $m \times \alpha$ transactions and each item must appears at least γ times.*

Note that in Proposition 1, the propositional formula $\Phi^{\alpha,\gamma,k}$ serves as a propositional encoding for the constraints involved in the closed k-RFPs extraction problem. In other words, this formula encapsulates the essential requirements for closed k-RFPs, guaranteeing a meaningful and reliable extraction of patterns that adhere to specified coverage and frequency constraints. It should be noted that when

$k = 0$, there is no relaxation. In fact, the 0-RFP patterns is identical to the classical closed patterns.

Next, we illustrate the importance of using k-RFP for clustering through the following example, utilizing the same dataset from Example 1.

Example 2. *Consider the same dataset $\mathcal{D}$ presented in Table 1. Let $k = 1$, $\alpha = 3$, $\gamma = 3$ and $P = \{a, b, c, d\}$ and $Q = \{e, f, g, h\}$, both being closed 1-RFPs. Then, the GDP representation of P and Q are correspondingly: $[\{a, b, c\}, \{a, b, d\}, \{a, c, d\}, \{b, c, d\}], [\{f, g, h\}, \{e, g, h\}, \{e, f, h\}, \{e, f, g\}]$ P and Q satisfies the four constraints. In fact, P and Q cover 4 transactions respectively T_1 to T_4 and T_5 to T_8 (Constraint (2)), and all transactions covered miss at most 1 item that exist in P and Q (Constraint (1)). Moreover, P and Q are closed (Constraint (3)). Furthermore, each item in P and Q respectively appears 3 times in each covered transactions respectively (Constraint (4)).*

Unlike traditional patterns, the clusters identified based on P and Q are $\{T_1, T_2, T_3, T_4\}$ and $\{T_5, T_6, T_7, T_8\}$, representing the desired best clusters that cover all the data. This demonstrates that the k-RFPs successfully addresses this issue, as elaborated in Subsection 3.1.

Our goal is to choose from the candidate k-RFPs by leveraging ILP models to obtain high-quality clusters. The detailed explanation of ILP models will be presented in Subsection 3.3.

3.3 Integer Linear Programming Models for Conceptual Clustering

In this section, we employ integer linear programming models to choose the optimal clusters. Before diving into the details, let us provide a formal definition of an ILP model. An ILP [35] can be described as a linear program, but it comes with an additional constraint: every variable involved must be an integer. The objective of an ILP problem is to maximize or minimize an objective function, subject to a set of linear constraints, while ensuring that all decision variables are integers.

The flexibility of ILP makes it applicable to a wide range of fields, including logistics, finance, manufacturing, and telecommunications, among others. Solving an ILP problem involves finding the optimal values for the decision variables that satisfy the constraints and optimize the objective function. Various algorithms, such as branch and bound [35], [6], [20] are employed to explore the solution space efficiently and identify the best combination of integer values for the variables.

In summary, ILP provides a powerful and versatile approach to solving optimization problems with discrete decision variables, making it a valuable tool in operational research and decision science. Denoting y as the vector of decision variables, where n represents the total number of integer variables. The objective coefficients, labeled as c_j for $1 \leq j \leq n$, signify the coefficients in the objective function. The matrix A has dimensions $h \times n$, where h is the number of constraints, containing coefficients for the constraints, and d is a $h \times 1$ vector representing the right-hand-side values of the constraints. Formally, the ILP problem takes the form:

$$\begin{aligned} \texttt{Maximize or Minimize} \quad & \mathbf{c}^T\mathbf{y} \\ \texttt{Subject to} \quad & \mathbf{A}\mathbf{y}(\leq, =\text{ or } \geq)\,\mathbf{d} \\ & \mathbf{y}_i \in \mathbb{Z}, i = 1, 2, \ldots, n \\ & \mathbf{c}_j \in \mathbb{R}, j = 1, 2, \ldots, n \end{aligned}$$

For our problem and following Ouali et al. [36], the conceptual clustering problem can be modeled into an ILP model. Then, we adopt the number of the covered transactions by a k-RFP as the objective function which should be maximized. The latter is under two constraints:

- The first constraint (1) states that each transaction T_i must be covered by exactly one k-RFP. However, the k-RFPs have higher cover than classical closed patterns. As a consequence, it can be difficult to satisfy the uniqueness constraint. To address this challenge, we limited the coverage by introducing a positive integer called `ILP-cover-threshold` σ where $\sigma < k$.
- The second constraint (2) limits the number of clusters to $\beta = \beta_0$ clusters.

Formally, our ILP model called **M1** is presented as follows:

$$\begin{aligned} \texttt{Maximize} \quad & \sum_{c \in C} v_c \cdot y_c \\ \texttt{Subject to} \quad (1) \quad & \sum_{c \in C} a_{T_i,c} \cdot y_c = 1, \quad \forall\, T_i \in \mathcal{D} \\ (2) \quad & \sum_{c \in C} y_c = \beta_0 \\ & y_c \in \{0, 1\}, c \in C \\ & i = 1, \ldots, m \end{aligned}$$

where $\mathcal{D}$ is a dataset with m transactions defined on a set of f items. Let C be the set of p closed k-RFPs. Let $a_{T_i,c}$ be an $m \times p$ binary matrix where $a_{T_i,c} = 1$ iff $|c \setminus T_i| \leq \sigma$ where $c \in C$. For classical patterns $a_{T_i,c} = 1$ iff $c \subseteq T_i$. Moreover, using p boolean variables y_c, where $y_c = 1$ iff the cluster represented by the closed k-RFP c belongs to the clustering. The objective function is defined by associating to each cluster c a value v_c reflecting the number of transactions covered by c, which should be maximized. Additionally, we aim to investigate the impact of relaxing the first constraint (1) of model M1 on the number of optimal solutions found. Subsequently, we propose a second model that allows for some transaction overlap. This model, referred to as **M2**, is presented as follows:

$$\begin{aligned} \texttt{Maximize} \quad & \sum_{c \in C} v_c \cdot y_c \\ \texttt{Subject to} \quad (1) \quad & \sum_{c \in C} a_{T_i,c} \cdot y_c \leq \theta, \quad \forall\, T_i \in \mathcal{D} \\ (2) \quad & \sum_{c \in C} y_c = \beta_0 \\ & y_c \in \{0, 1\}, c \in C \\ & i = 1, \ldots, m \end{aligned}$$

M2 has the same objective function, variables, and constraint (2) as M1. The only difference is that constraint (1) in M2 specifies that each transaction is covered by at most θ relaxed patterns.

We have theoretically the k-RFP model, and the corresponding ILP models M1 and M2. To demonstrate the practical applicability of our approach, we will detail our experiments in Section 4.

4 Experimental Evaluation

To demonstrate the efficiency of our proposed approach, we conducted an empirical evaluation on various well-known real-world datasets [1] as presented in Table 2.

[1] The datasets were collected from the UCI repository are available at: https://dtai.cs.kuleuven.be/CP4IM/datasets/.

Table 2. Real-world datasets caracteristics

$\mathcal{D}$	#Transactions	#Items	Density (%)
Lymph	148	68	40
Mushroom	8124	119	18
Primary-Tumor	336	31	48
Soybean	630	50	32
Tic-tac-toe	958	27	33
Vote	435	48	33
Zoo-1	101	36	44

First, we computed the classical closed patterns (CCP), which were identified for $k = 0$ and then the k-RFPs ($k \geq 1$). It should be noted that, the number of enumerated patterns will increase due to the relaxation requirement. Therefore, we set $k = 1$ to prevent having too many patterns.

To enumerate our k-RFPs, we used the well-known satisfiability solver MiniSAT [11] implemented in C++. Our implementation includes slight modifications to MiniSAT to enumerate all models. We followed the approach of [10] and we considered a non-blocking models approach. The restart strategy is then disabled and a simple backtracking is performed each time a model is found. Moreover, the clause learning component is also disabled for efficiency reasons. Finally for variables selection strategy, we started by assigning variables representing items. Those of transactions are then propagated accordingly. Subsequently, leveraging the extracted patterns, we identified the set of clusters using ILP models described in Subsection 3.3. Our implementation was carried out in Python v3.9 using the `Pulp` framework. The implementation of the modified solver and ILP models is available in [4]. For our ILP models we fixed σ to 0. Moreover, we adjusted the timeout to 7 hours. All experiments were carried out on a MacBook Air with 16 GB of RAM.

Subsequently, we evaluated the effectiveness of our novel patterns by initially applying our k-RFPs to the model M1. Specifically, we extracted the k-RFPs by varying the minimum support α from 10% to 40%. When k-RFPs extracted with a fixed $\alpha = 10\%$ were used for M1, we called the Conceptual Clustering Approach based on k-RFP (`CCA-`k`-RFP-M1`). This approach was compared with all classical closed patterns on M1, (`CCP-M1`). Additionally, we compared `CCA-`k`-RFP-M1` with various other disjoint clustering methods. Moreover, we applied our k-RFPs extracted with $\alpha = 10\%$ to the model M2, we called `CCA-`k`-RFP-M2`, which is compared with all classical closed patterns on the model M2, referred to as `CCP-M2`, by varying the θ parameter from 2 to 5 in order to observe the impact of relaxing the first constraint (1) on the number of optimal solutions found. In addition, we compared `CCA-`k`-RFP-M2` with another well-known overlapping clustering method, `Neo-`k`-means`.

All these comparisons are conducted in terms of the running time required for searching the optimal solution in the ILP models, the number of the found optimal solutions and the quality of the identified clusters.

To evaluate the quality of a clustering, we performed a measure called the Intra-Cluster Similarity (ICS). We adopted the Jaccard similarity measure, computed as follows: Given two transactions T_i and T_j where $i \neq j$ and $i, j \in [1, m]$ we have s: $\mathcal{D} \times \mathcal{D} \mapsto [0, 1]$, $s(T_i, T_j) = \frac{|T_i \cap T_j|}{|T_i \cup T_j|}$. Then:

$$ICS(c_1, \ldots, c_\beta) = \frac{1}{2} \sum_{1 \leq r \leq \beta} \Big(\sum_{T_i, T_j \in c_r} s(T_i, T_j) \Big)$$

For M1, we varied the minimum support threshold α and the minimum item frequency γ from 10% to 40%. For efficiency reasons, we set $\gamma = \alpha$ for both classical closed patterns and k-RFPs. For each variation of α, we varied the desired number of clusters β from 3 to 30.

For M2, we varied θ from 2 to 5, and for each variation of θ, the number of desired clusters β is varied from 3 to 30 because our goal is to observe the impact of relaxing constraint (1) on the number of optimal solutions found when using our novel patterns. For k-RFPs, $\gamma = \alpha = 10\%$, while for classical patterns, we extracted all patterns ($\alpha = 0\%$).

To compare with other approaches, we used the following settings: For the model M1, we fixed $\alpha = 10\%$ and $\beta = 30$. For M2, α and β were kept the same as in M1, while the relaxation parameter $\theta = 2$.

It is important to note that the purpose of comparing our approach with various other clustering methods is to demonstrate its competitiveness in terms of clustering quality.

Analysis of the number of found optimal solutions for M1 and M2. Table 3 illustrate the obtained results for both classical and relaxed patterns applied to model M1. Notably, our consideration is limited to optimal solutions found using ILP. As expected, increasing the minimum support threshold α decreases the number of both k-RFPs and classical closed patterns. However, the number of k-RFPs is higher than that of classical patterns, due to the relaxation. According to the obtained results, we observe that for classical closed patterns, the number of found solutions for many datasets is at most 8. This is the case for `Lymph` data. Indeed, the solutions are found within the fixed timeout for low values of β i.e., from 3 to a maximum of 10, especially for α close to 40%. This is the case for the `Vote` dataset. However, using relaxed patterns, our approach allow to find solutions for a wide range of β values showing the relevance of k-RFPs to reach the clustering goal and demonstrating that k-RFPs surpasses the one of classical patterns. Specifically, the relaxed patterns allows to find solutions for values of β ranged from 3 to 30 (up to 28 solutions) for many datasets e.g., `Zoo-1`, `Vote` and `Tic-tac-toe`.

Table 4 presents the comparison between `CCA-`k`-RFP-M2` and `CCP-M2`. The results demonstrate that our method achieves optimal solutions for all variations of θ and β across all datasets. However, `CCP-M2` fails to extract optimal solutions for certain variations of β. This is the case for `Lymph`, `Mushroom`, and `Tic-tac-toe` datasets.

These results underscore how relaxing classical patterns enhances the likelihood of solution discovery for both M1 and M2 models.

Analysis of running time for M1 and M2. Table 3 presents the average CPU time required to find the optimal clustering solution for M1. The average running time is calculated only for configurations where optimal solutions are found, it is determined by summing the running times for each β variation and then dividing by the number of these variations. As we can remark, the average running time for classical patterns is faster than for k-RFP. This outcome aligns with expectations, as the number of classical closed itemsets is relatively smaller compared to k-RFPs. Moreover, It is known that the average running time is calculated only for configurations where optimal solutions are found. Since the number of optimal solutions is higher when using k-RFPs compared to classical closed patterns, this contributes to the increased running time. Furthermore, as α increases, the average CPU time decreases.

Table 4 shows that `CCA-`k`-RFP-M2` outperforms `CCP-M2` in terms of solving time across various θ values and datasets, including `Primary-Tumor`, `Soybean`, `Tic-tac-toe`, and `Vote`. However, exceptions were observed for `Lymph`, `Zoo-1`, and `Mushroom`

Table 3. k-RFP for M1 vs. Classical Closed pattern for M1

$\mathcal{D}$	α	k-RFP			Classical Closed pattern		
		#k-RFP	#Found sol	Average time (s)	#CCP	#Found sol	Average time (s)
Lymph	10%	3605378	**27**	3760.40	51862	8	**62.85**
	20%	759630	**27**	415.81	13934	2	**2.37**
	30%	202602	**27**	74.12	4910	1	**0.92**
	40%	60470	0	-	2058	0	-
Mushroom	10%	128962	**27**	930.68	3287	6	**34.21**
	20%	19712	**27**	74.59	817	1	**6.03**
	30%	4055	0	-	293	**1**	**3.06**
	40%	1135	0	-	107	0	-
Primary-Tumor	10%	256991	**27**	42.71	32183	7	**19.70**
	20%	76081	**27**	13.89	9891	2	**6**
	30%	30372	**27**	8.88	3614	1	**1.42**
	40%	14778	0	-	1382	0	-
Soybean	10%	69191	**27**	16.19	2907	6	**2.57**
	20%	11900	0	-	844	**2**	**0.53**
	30%	3383	0	-	380	0	-
	40%	1484	0	-	205	0	-
Tic-tac-toe	10%	4479	**28**	19.94	191	2	**0.17**
	20%	811	**28**	1.45	26	1	**0.10**
	30%	171	0	-	18	0	-
	40%	15	0	-	5	0	-
Vote	10%	280386	**28**	67.38	37399	3	**14.93**
	20%	34098	0	-	7227	0	-
	30%	6606	0	-	658	0	-
	40%	693	0	-	79	0	-
Zoo-1	10%	92711	**28**	4.71	3291	7	**0.32**
	20%	35081	**28**	2.02	1743	2	**0.23**
	30%	12614	**28**	0.86	818	1	**0.14**
	40%	3555	**28**	**0.28**	316	0	-

Table 4. CCA-k-RFP-M2 vs. CCP-M2

$\mathcal{D}$	θ	CCA-k-RFP-M2		CCP-M2	
		#Found sol	Average time (s)	#Found sol	Average time (s)
Lymph	2	28	4156.28	28	**2536.88**
	3	**28**	3519.44	26	**351.28**
	4	**28**	6115.981	26	**623.40**
	5	**28**	3998.53	24	**218.67**
Mushroom	2	**28**	**355.77**	2	15244.37
	3	**28**	362.99	1	**195.29**
	4	**28**	355.79	1	**187.84**
	5	**28**	332.93	1	**196.36**
Primary-Tumor	2	28	**47.96**	28	198.12
	3	28	**48.54**	28	183.98
	4	28	**59.25**	28	159.136
	5	28	**52.73**	28	123.17
Soybean	2	28	**11.97**	28	137.062
	3	28	**11.72**	28	100.11
	4	28	**12.72**	28	80.66
	5	28	**10.68**	28	73.489
Tic-tac-toe	2	**28**	**27.01**	27	1693.54
	3	**28**	**22.46**	27	447.67
	4	**28**	**19.60**	25	227.98
	5	**28**	**16.24**	24	221.14
Vote	2	28	**77.77**	28	2716.59
	3	28	**103.17**	28	2660.54
	4	28	**531.33**	28	1436.24
	5	28	**240.68**	28	963.59
Zoo-1	2	28	5.12	28	**0.61**
	3	28	5.27	28	**0.53**
	4	28	5.25	28	**0.54**
	5	28	5.24	28	**0.53**

(with θ ranging from 3 to 5). This is due to two factors: the number of patterns found and the order in which the SAT solver identifies them, both of which can affect the ILP solver's runtime.

CCA-k-RFP-M1 vs. CCP-M1, CCA-k-RFP-M1 vs. other methods and CCA-k-RFP-M2 vs. Neo-k-Means. The results of Table 5 show that our approach outperforms `CCP-M1` in terms of cluster quality across all datasets. To quantify this superiority, we specifically examined the ICS quality values for both approaches. The results revealed that the ICS values achieved by our approach were consistently higher, with the improvement ranging from approximately

Table 5. CCA-k-RFP-M1 vs. CCP-M1

$\mathcal{D}$	CCA-k-RFP-M1			CCP-M1		
	$k=1, \beta=30, \alpha=10\%$			$\beta=30$		
	#k-RFP	ICS	Time (s)	#CCP	ICS	Time (s)
Lymph	3605378	**3723.17**	3305.40	154220	277.93	**270.67**
Mushroom	128962	**5652568.44**	**932.69**	221524	-	-
Primary-	256991	**15984.25**	**44.11**	87230	1895.89	241.95
Soybean	69191	**40133.02**	**14.23**	31759	10795.60	87.50
Tic-tac-toe	4479	**47165.55**	**19.52**	42711	29278.23	1466.06
Vote	280386	**16992.90**	**68.75**	227031	8563.77	1268.51
Zoo-1	92711	**768.37**	4.74	4567	267.79	**0.422**

Table 6. CCA-k-RFP-M1 vs. Other Clustering methods for $\beta = 30$.

$\mathcal{D}$	CCA-k-RFP-M1		k-Means		BIRCH		SPECTRAL		Aggl-Clust	
	ICS	Time (s)	ICS	Time (s)	ICS	Time (s)	ICS	Time (s)	ICS	Time (s)
Lymph	**3723.17**	3305.40	252.80	0.45	257.89	0.04	452.48	0.429	250.86	**0.008**
Mushroom	**5652568.44**	932.69	883931.50	**3.17**	926729.75	15.90	1442391.09	100.98	1072855.32	11.32
Primary-Tumor	**15984.25**	44.11	1745.32	0.47	2030.70	0.06	5632.47	0.76	1792.95	**0.014**
Soybean	**40133.02**	14.23	5067.89	0.64	5433.03	0.182	17066	1.14	5040.35	**0.04**
Tic-tac-toe	**47165.55**	19.52	7751.78	0.89	7049.87	0.32	7353.16	2.07	7059.29	**0.08**
Vote	**16992.90**	68.75	2794.59	0.51	3189.58	0.12	14807.83	0.86	2533.51	**0.02**
Zoo-1	**768.37**	4.74	218.60	0.39	248.17	0.02	467.03	0.23	164.77	**0.005**

Table 7. CCA-k-RFP-M2 vs. Neo-k-Means for $\beta = 30$ and $\theta = 2$.

$\mathcal{D}$	CCA-k-RFP-M2		Neo-k-Means	
	ICS	Time (s)	ICS	Time (s)
Lymph	**7470.67**	3288.04	461	**0.21**
Mushroom	**12259500.13**	332.13	1967290.99	**7.36**
Primary-Tumor	**36752.61**	50.74	2965.39	**0.16**
Soybean	**71741.53**	13.12	6344.39	**0.35**
Tic-tac-toe	**92792.95**	34.43	10917.74	**0.58**
Vote	**31761.66**	77.37	6083.92	**0.27**
Zoo-1	**1657.55**	5.01	285.84	**0.06**

1.6 to 13 times compared to those achieved by CCP−M1. This substantial enhancement in cluster quality provides a clear demonstration of the efficacy of our technique when contrasted with the performance of CCP−M1. The CPU running time of CCA−k−RFP−M1 is generally faster than that of CCP−M1 across multiple datasets. However, exceptions are observed for Lymph and Zoo−1, where the number of closed patterns is lower than the number of k-RFPs. This is due to the same two factors mentioned earlier. For instances like Primary−Tumor, Soybean, Mushroom, and Vote, where the number of k-RFPs exceeds that of classical patterns, CCA−k−RFP−M1 still faster than CCP−M1.

In Table 6, our method surpasses the other approaches in terms of quality across all datasets. However, in terms of runtime, Aggl−Clust is faster than the other approaches except for Mushroom, were k−Means is faster. This is because our approach is configured to search only for optimal solutions, which takes much time to find the best quality and especially when the number of patterns found is high. Therefore, in terms of quality, our approach is the best.

Table 7 present the results of the comparison between our approach and Neo−k−Means. In the quality side our method overpasses Neo−k−Means across all datasets with a high difference. However, on the running time side Neo−k−Means is the fastest. This is because, as explained above, our approach is configured to search only for optimal solutions.

The results obtained are intriguing and demonstrate the effectiveness of our method in terms of both the number of optimal solutions found and the quality of the clusters produced. This is particularly significant in terms of cluster quality when dealing with a large number of clusters. Consequently, the findings suggest that relaxing constraints on closed patterns can potentially enhance cluster quality.

5 Conclusion and Forthcoming Work

In this paper, we presented a novel approach to conceptual clustering based on a relaxation of closed frequent itemsets named k-RFPs defined by relaxing the support of itemsets. In fact, this innovative pattern model is a notable extension of classical closed patterns, allowing the flexibility to accommodate the nonappearance of up to k items in each transaction that support the pattern. Our approach is divided into two phases. First, we applied the SAT encoding previously used in the context of association rule mining to enumerate the k-RFPs. Second, using such patterns, ILP models are used to identify the best clusters. Indeed, we proposed two ILP models: the first one we called M1, which represents the conceptual clustering problem, and the second one, M2, is proposed to observe the impact of relaxing the first constraint of M1 on the number of optimal solutions found. Third, experimental evaluation was conducted to assess the effectiveness of the proposed approach compared to classical closed itemsets. The experiments show that optimal clustering can be reached for high values of clusters number while classical closed itemsets failed.

As part of our future work, firstly, we plan to enhance our symbolic encoding for k-RFPs to tackle scalability issues. Indeed, for $k > 1$, the solver has the potential to generate numerous patterns, leading to the increase of running time, especially when dealing with large datasets with high density. For this purpose, parallelization and decomposition can be used to reach this goal. Secondly, we plan to investigate the use of k-RFPs for community detection in social networks. Social network datasets can be modeled as undirected graphs, which can be represented similarly to transactional datasets. Thirdly, we plan to improve the runtime of the ILP solver. We have observed that the order in which patterns are used as input for the ILP model can influence the solving time.

Acknowledgement

This research has received support from the European Union's Horizon research and innovation programme under the MSCA-SE (Marie Skłodowska-Curie Actions Staff Exchange) grant agreement 101086252; Call: HORIZON-MSCA-2021-SE-01; Project title: STARWARS (STormwAteR and WastewAteR networkS heterogeneous data AI-driven management).

References

[1] M. Ackerman and S. Ben-David. Discerning linkage-based algorithms among hierarchical clustering methods. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[2] M. R. Ackermann, J. Blömer, D. Kuntze, and C. Sohler. Analysis of agglomerative clustering. *Algorithmica*, 69:184–215, 2014.

[3] O. Bailleux, Y. Boufkhad, and O. Roussel. A translation of pseudo-boolean constraints to sat. *Journal on Satisfiability, Boolean Modeling and Computation*, 2(1-4):191–200, 2006.

[4] M. BEN HASSINE. Conceptual clustering based on relaxed patterns, Aug. 2024. URL https://doi.org/10.5281/zenodo.13285611.

[5] A. Boudane, S. Jabbour, B. Raddaoui, and L. Sais. Efficient sat-based encodings of conditional cardinality constraints. In *LPAR*, pages 181–195, 2018.

[6] S. Ceria, C. Cordier, H. Marchand, and L. A. Wolsey. Cutting planes for integer programs with general integer variables. *Mathematical programming*, 81:201–214, 1998.

[7] T.-B.-H. Dao, C.-T. Kuo, S. Ravi, C. Vrain, and I. Davidson. Descriptive clustering: Ilp and cp formulations with applications. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 1263–1269, 2018.

[8] L. De Raedt, T. Guns, and S. Nijssen. Constraint programming for itemset mining. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 204–212, 2008.

[9] I. O. Dlala, S. Jabbour, B. Raddaoui, and L. Sais. A parallel sat-based framework for closed frequent itemsets mining. In *International Conference on Principles and Practice of Constraint Programming*, pages 570–587, 2018.

[10] I. O. Dlala, S. Jabbour, B. Raddaoui, and L. Sais. A parallel sat-based framework for closed frequent itemsets mining. In J. N. Hooker, editor, *Principles and Practice of Constraint Programming - 24th International Conference*, pages 570–587. Springer, 2018.

[11] N. Eén and N. Sörensson. An extensible sat-solver. In *International conference on theory and applications of satisfiability testing*, pages 502–518. Springer, 2003.

[12] N. Eén and N. Sörensson. Translating pseudo-boolean constraints into sat. *Journal on Satisfiability, Boolean Modeling and Computation*, 2 (1-4):1–26, 2006.

[13] B. S. Everitt, S. Landau, M. Leese, and D. Stahl. *Cluster analysis*. John Wiley & Sons, 2011.

[14] D. H. FISHER. Conceptual clustering, learning from examples, and inference. In *Proceedings of the Fourth International Workshop on Machine Learning*, pages 38–49. Elsevier, 1987.

[15] J. H. Gennari. A survey of clustering methods. 1989.

[16] T. Guns, S. Nijssen, and L. De Raedt. k-pattern set mining under constraints. *IEEE Transactions on Knowledge and Data Engineering*, 25 (2):402–418, 2013. doi: 10.1109/TKDE.2011.204.

[17] T. Guns, A. Dries, S. Nijssen, G. Tack, and L. De Raedt. Miningzinc: A declarative framework for constraint-based mining. *Artificial Intelligence*, 244:6–29, 2017.

[18] M. B. Hassine, S. Jabbour, M. Kmimech, B. Raddaoui, and M. Graiet. A non-overlapping community detection approach based on α-structural similarity. In *International Conference on Big Data Analytics and Knowledge Discovery*, pages 197–211. Springer, 2023.

[19] A. Hidouri, S. Jabbour, B. Raddaoui, and B. B. Yaghlane. Mining closed high utility itemsets based on propositional satisfiability. *Data Knowl. Eng.*, 136:101927, 2021.

[20] F. S. Hillier. *Introduction to operations research*. McGrawHill, 2001.

[21] Y. Izza, S. Jabbour, B. Raddaoui, and A. Boudane. On the enumeration of association rules: A decomposition-based approach. In *International Joint Conference on Artificial Intelligence*, pages 1265–1271, 2020.

[22] S. Jabbour, L. Saïs, and Y. Salhi. A pigeon-hole based encoding of cardinality constraints. In *International Symposium on Artificial Intelligence and Mathematics, ISAIM 2014, Fort Lauderdale, FL, USA, January 6-8, 2014*, 2014.

[23] S. Jabbour, F. E. Mana, I. O. Dlala, B. Raddaoui, and L. Sais. On maximal frequent itemsets mining with constraints. In *International Conference on Principles and Practice of Constraint Programming*, pages 554–569, 2018.

[24] S. Jabbour, N. Mhadbhi, B. Raddaoui, and L. Sais. Triangle-driven community detection in large graphs using propositional satisfiability. In *2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA)*, pages 437–444. IEEE, 2018.

[25] S. Jabbour, N. Mhadhbi, B. Raddaoui, and L. Sais. Sat-based models for overlapping community detection in networks. *Computing*, 102(5): 1275–1299, 2020.

[26] S. Jabbour, N. Mhadhbi, B. Raddaoui, and L. Sais. A declarative framework for maximal k-plex enumeration problems. In *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS*, 2022.

[27] S. Jabbour, B. Raddaoui, and L. Sais. A symbolic approach to computing disjunctive association rules from data. In *Thirty-Second International Joint Conference on Artificial Intelligence {IJCAI-23}*, pages 2133–2141. International Joint Conferences on Artificial Intelligence Organization, 2023.

[28] X. Jin and J. Han. K-means clustering. *Encyclopedia of machine learning*, pages 563–564, 2011.

[29] L. Kaufman and P. J. Rousseeuw. *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 2009.

[30] M. E. A. Laghzaoui and Y. Lebbah. A constraint programming approach for quantitative frequent pattern mining. *International Journal of Data Mining, Modelling and Management*, 15(3):297–311, 2023.

[31] M. Lai, L. Cao, H. Lu, Q. Ha, L. Li, J. Hossain, and P. Kennedy. An unsupervised hierarchical clustering approach to improve hopfield retrieval accuracy. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2023.

[32] J.-P. Métivier, P. Boizumault, B. Crémilleux, M. Khiari, and S. Loudni. Constrained clustering using sat. In *Advances in Intelligent Data Analysis XI*, pages 207–218, 2012.

[33] R. S. Michalski. *Knowledge acquisition through conceptual clustering: A theoretical framework and an algorithm for partitioning data into conjunctive concepts*. Number 1026. Department of Computer Science, University of Illinois at Urbana-Champaign . . . , 1980.

[34] A. A. Nanavati, K. P. Chitrapura, S. Joshi, and R. Krishnapuram. Mining generalised disjunctive association rules. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 482–489, 2001.

[35] G. L. Nemhauser and L. A. Wolsey. Integer and combinatorial optimization john wiley & sons. *New York*, 118, 1988.

[36] A. Ouali, S. Loudni, Y. Lebbah, P. Boizumault, A. Zimmermann, and L. Loukil. Efficiently finding conceptual clustering models with integer linear programming. In *25th International Joint Conferences on Artificial Intelligence*, 2016.

[37] A. Ouali, A. Zimmermann, S. Loudni, Y. Lebbah, B. Crémilleux, P. Boizumault, and L. Loukil. Integer linear programming for pattern set mining; with an application to tiling. In *Advances in Knowledge Discovery and Data Mining: 21st Pacific-Asia Conference, PAKDD 2017*, pages 286–299. Springer, 2017.

[38] S. M. Savaresi, D. L. Boley, S. Bittanti, and G. Gazzaniga. Cluster selection in divisive clustering algorithms. In *Proceedings of the 2002 SIAM International Conference on Data Mining*, pages 299–314. SIAM, 2002.

[39] Shi. Multiclass spectral clustering. In *Proceedings ninth IEEE international conference on computer vision*, pages 313–319. IEEE, 2003.

[40] C. Sinz. Towards an optimal cnf encoding of boolean cardinality constraints. In *International conference on principles and practice of constraint programming*, pages 827–831. Springer, 2005.

[41] J. H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.

[42] J. J. Whang, I. S. Dhillon, and D. F. Gleich. Non-exhaustive, overlapping k-means. In *Proceedings of the 2015 SIAM international conference on data mining*, pages 936–944. SIAM, 2015.

[43] T. Xiong, S. Wang, A. Mayers, and E. Monga. Dhcc: Divisive hierarchical clustering of categorical data. *Data Mining and Knowledge Discovery*, 24:103–135, 2012.

[44] T. Zhang, R. Ramakrishnan, and M. Livny. Birch: an efficient data clustering method for very large databases. *ACM sigmod record*, 25 (2):103–114, 1996.