Supplementary Material

This Supplementary Material is a companion document to our main research paper, providing additional depth and clarity to key aspects of our work.
Section 1 offers detailed proofs of the propositions presented in the paper, ensuring rigorous theoretical validation. **Section 2** explores the sampling capabilities of our mining model through a comparative analysis of extracted patterns using PCA and some additional clustering figures. **Section 3** describes the solver's search mechanism in detail, formalizing its backtracking approach to efficiently navigate the search space and generate diverse, meaningful results. **Section 4** introduces the `ClosedPattern` global constraint. Finally, **Section 5** provides an accurate calculation method for the joint entropy of the extracted pattern set, ensuring precise measurement and evaluation of our results.

1 Proofs of Propositions

This section provides comprehensive proof of the propositions presented in the main body of our research paper.

1.1 Proof of proposition 1 (Monotonicity of the Overlap measure)

Proof. The proof of this proposition is provided through the following counter-example 1.

□

Example 1. *Consider the transaction database $\mathcal{D}$ shown in Table 1 of our paper. Let $H = \{A, E\}$ (see the search tree given in Fig. 3) be a pattern member of the history $\mathcal{H}$ extracted from $\mathcal{D}$. Now, let's consider a set of partial assignment $\sigma^+ = \{A\} \subseteq \{A, C\} \subseteq \{A, C, D\} \subseteq \{A, C, D, E\}$. The overlap coefficients of these sets w.r.t. H are respectively:* 1, $\frac{1}{3}$, $\frac{1}{3}$, *and* 1 *(see Figure 1). This figure illustrates the overlap coefficient of sets with inclusion assignments σ^+. Through the analysis of computed overlap coefficients, we can observe that the overlap coefficient measure doesn't exhibit a monotonic or anti-monotonic behavior in the context of set inclusion. This illustrative counterexample strongly reinforces our argument that the overlap coefficient measure doesn't conform to monotonicity or anti-monotonicity when applied to sets with inclusion assignments.*

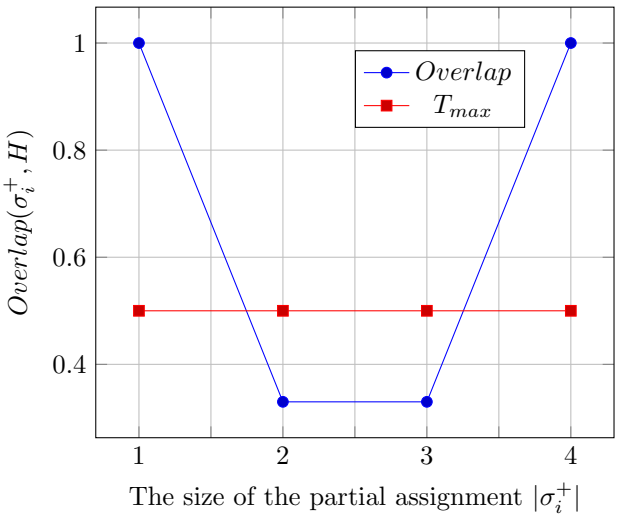


Fig. 1: The overlap coefficient of an increasing set of partial assignments σ_i^+ w.r.t. a pattern H, i.e. $Overlap(\sigma_i^+, H), i = 1..4$, where $\sigma_i^+ \subset \sigma_j^+, \forall i < j$.

1.2 Proof of Lemma 1 (Proper cover)

Proof. As $P \subseteq Q$ We get:

$$\begin{aligned} |\mathcal{V}_{\mathcal{D}}(P)| &\geq |\mathcal{V}_{\mathcal{D}}(Q)| \\ \Rightarrow |\mathcal{V}_{\mathcal{D}}(P) \cup \mathcal{V}_{\mathcal{D}}(H)| &\geq |\mathcal{V}_{\mathcal{D}}(Q) \cup \mathcal{V}_{\mathcal{D}}(H)| \end{aligned}$$

Since $|\mathcal{V}_{\mathcal{D}}(P) \cup \mathcal{V}_{\mathcal{D}}(H)| = |\mathcal{V}_H^{pr}(P)| + |\mathcal{V}_{\mathcal{D}}(H)|$, we get:

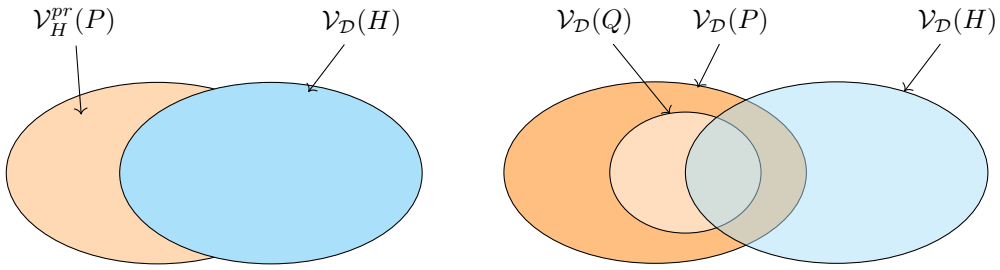
$$\begin{aligned} &\Rightarrow |\mathcal{V}_H^{pr}(P)| + |\mathcal{V}_{\mathcal{D}}(H)| \geq |\mathcal{V}_H^{pr}(Q)| + |\mathcal{V}_{\mathcal{D}}(H)| \\ &\Rightarrow |\mathcal{V}_H^{pr}(P)| \geq |\mathcal{V}_H^{pr}(Q)| \end{aligned}$$

□

1.3 Proof of Theorem 1 (Lower Bound)

Proof. $\forall H \in \mathcal{H}$ we have:

$$\begin{aligned} \mathcal{V}_H^{pr}(P) &= \mathcal{V}_{\mathcal{D}}(P) \setminus \{\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)\} \\ \Rightarrow |\mathcal{V}_H^{pr}(P)| &= |\mathcal{V}_{\mathcal{D}}(P)| - |\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)| \\ \Rightarrow |\mathcal{V}_{\mathcal{D}}(P)| &= |\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)| + |\mathcal{V}_H^{pr}(P)| \end{aligned}$$



(a) Proper cover of P w.r.t. H (b) Proper cover s.t. $P \subseteq Q$

Fig. 2: Proper cover $\mathcal{V}_H^{pr}$ and cover $\mathcal{V}_\mathcal{D}$ of P, Q w.r.t H.

We also have:

$$
\begin{aligned}
& |\mathcal{V}_\mathcal{D}(P)| \geq \theta \\
\Rightarrow\ & |\mathcal{V}_\mathcal{D}(P) \cap \mathcal{V}_\mathcal{D}(H)| + |\mathcal{V}_H^{pr}(P)| \geq \theta \\
\Rightarrow\ & |\mathcal{V}_\mathcal{D}(P) \cap \mathcal{V}_\mathcal{D}(H)| \geq \theta - |\mathcal{V}_H^{pr}(P)| \\
\Rightarrow\ & \frac{|\mathcal{V}_\mathcal{D}(P) \cap \mathcal{V}_\mathcal{D}(H)|}{\min(|\mathcal{V}_\mathcal{D}(P)|, |\mathcal{V}_\mathcal{D}(H)|)} \geq \frac{\theta - |\mathcal{V}_H^{pr}(P)|}{\min(|\mathcal{V}_\mathcal{D}(P)|, |\mathcal{V}_\mathcal{D}(H)|)} \\
\Rightarrow\ & \mathit{Overlap}(P, H) \geq \frac{\theta - |\mathcal{V}_H^{pr}(P)|}{\min(|\mathcal{V}_\mathcal{D}(P)|, |\mathcal{V}_\mathcal{D}(H)|)}
\end{aligned}
$$

Here, we distinguish two cases:

$$
\begin{aligned}
\theta \geq |\mathcal{V}_H^{pr}(P)| &\Rightarrow \theta - |\mathcal{V}_H^{pr}(P)| \geq 0 \ \vee \\
\theta < |\mathcal{V}_H^{pr}(P)| &\Rightarrow \theta - |\mathcal{V}_H^{pr}(P)| < 0 \\
&\Rightarrow LB_{OC}(P, H) = \max\left(\frac{\theta - |\mathcal{V}_H^{pr}(P)|}{\min(|\mathcal{V}_\mathcal{D}(P)|, |\mathcal{V}_\mathcal{D}(H)|)}, 0\right)
\end{aligned}
$$

□

1.4 Proof of proposition 2 (Monotonicity of LB_{OC})

Proof. Since $|\mathcal{V}_H^{pr}(P)| \geq |\mathcal{V}_H^{pr}(Q)|$ (see Lemma 1), we get

$$
\begin{aligned}
& -|\mathcal{V}_H^{pr}(P)| \leq -|\mathcal{V}_H^{pr}(Q)| \\
\Rightarrow\ & \theta - |\mathcal{V}_H^{pr}(P)| \leq \theta - |\mathcal{V}_H^{pr}(Q)| \\
\Rightarrow\ & \frac{\theta - |\mathcal{V}_H^{pr}(P)|}{min(|\mathcal{V}_\mathcal{D}(P)|, |\mathcal{V}_\mathcal{D}(H)|)} \leq \frac{\theta - |\mathcal{V}_H^{pr}(Q)|}{min(|\mathcal{V}_\mathcal{D}(P)|, |\mathcal{V}_\mathcal{D}(H)|)}
\end{aligned}
$$

$$
\begin{aligned}
&\Rightarrow \frac{\theta - |\mathcal{V}_H^{pr}(P)|}{min(|\mathcal{V}_{\mathcal{D}}(P)|, |\mathcal{V}_{\mathcal{D}}(H)|)} \leq \frac{\theta - |\mathcal{V}_H^{pr}(Q)|}{min(|\mathcal{V}_{\mathcal{D}}(Q)|, |\mathcal{V}_{\mathcal{D}}(H)|)} \\
&\Rightarrow LB_{OC}(P,\, H) \leq LB_{OC}(Q,\, H)
\end{aligned}
$$

□

1.5 Proof of Theorem 2 (Upper Bound)

Proof.

$$
\begin{aligned}
&|\mathcal{V}_{\mathcal{D}}(P)| \geq \theta \wedge |\mathcal{V}_{\mathcal{D}}(H)| \geq \theta \\
&\Rightarrow \min(|\mathcal{V}_{\mathcal{D}}(P)|, |\mathcal{V}_{\mathcal{D}}(H)|) \geq \theta \\
&\Rightarrow \frac{1}{\min(|\mathcal{V}_{\mathcal{D}}(P)|, |\mathcal{V}_{\mathcal{D}}(H)|)} \leq \frac{1}{\theta} \\
&\Rightarrow \frac{|\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)|}{\min(|\mathcal{V}_{\mathcal{D}}(P)|, |\mathcal{V}_{\mathcal{D}}(H)|)} \leq \frac{|\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)|}{\theta} \\
&\Rightarrow Overlap(P,\, H) \leq \frac{|\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)|}{\theta}
\end{aligned}
$$

At this stage, we distinguish two cases:

$$
\begin{aligned}
&|\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)| \geq \theta \ \vee \ |\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)| \leq \theta \\
&\Rightarrow \frac{|\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)|}{\theta} \geq \frac{\theta}{\theta} = 1 \ \vee \ \frac{|\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)|}{\theta} \leq \frac{\theta}{\theta} = 1 \\
&\Rightarrow UB_{OC}(P,\, H) = \frac{\min(|\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)|, \theta)}{\theta} \\
&\Rightarrow UB_{OC}(P,\, H) = \min\left(\frac{|\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)|}{\theta}, 1\right)
\end{aligned}
$$

□

1.6 Proof of Proposition 3 (Anti-monotonicity of UB_{OC})

Proof.

$$
\begin{aligned}
&P \subseteq Q \\
&\Rightarrow |\mathcal{V}_{\mathcal{D}}(P)| \geq |\mathcal{V}_{\mathcal{D}}(Q)| \\
&\Rightarrow |\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)| \geq |\mathcal{V}_{\mathcal{D}}(Q) \cap \mathcal{V}_{\mathcal{D}}(H)| \\
&\Rightarrow \frac{|\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)|}{\theta} \geq \frac{|\mathcal{V}_{\mathcal{D}}(Q) \cap \mathcal{V}_{\mathcal{D}}(H)|}{\theta} \\
&\Rightarrow \min\left(\frac{|\mathcal{V}_{\mathcal{D}}(P) \cap \mathcal{V}_{\mathcal{D}}(H)|}{\theta}, 1\right) \geq \min\left(\frac{|\mathcal{V}_{\mathcal{D}}(Q) \cap \mathcal{V}_{\mathcal{D}}(H)|}{\theta}, 1\right) \\
&\Rightarrow UB_{OC}(P,\, H) \geq UB_{OC}(Q,\, H)
\end{aligned}
$$

□

1.7 Proof of Proposition 5 (`OverlapDiv` filtering rules)

Proof. The proof of filtering rule (**R1**) follows from Proposition 3 in our paper. Now, we give the proof of the second filtering rule (**R2**). Let $P_1 = \sigma^+ \cup \{k\}$ and $P_2 = \sigma^+ \cup \{i\}$ such that $i \in \sigma^*$ (i.e. free item) and $k \in \sigma^-$ (i.e. filtered item), and $H \in \mathcal{H}$:

$$
\begin{array}{lll}
\text{We have} & \mathcal{V}_{\mathcal{D}}(\sigma^+ \cup \{i\}) \subseteq \mathcal{V}_{\mathcal{D}}(\sigma^+ \cup \{k\}) & \\
\Rightarrow & \mathcal{V}_{\mathcal{D}}(P_2) \subseteq \mathcal{V}_{\mathcal{D}}(P_1) & \\
\Rightarrow & P_1 \subseteq P_2 & \\
\Rightarrow & LB_{OC}(P_1, H) \leq LB_{OC}(P_2, H) & \text{(according to Proposition 3 in our paper)} \\
k \in \sigma^- \quad \Rightarrow & LB_{OC}(P_1, H) > T_{\max}, \text{ hence } LB_{OC}(P_2, H) > T_{\max} &
\end{array}
$$

□

1.8 Proof of Proposition 6 (Time Complexity Analysis)

Proof. Line 3 of Algorithm 1 runs in $O(m \times |\mathcal{H}|)$ in the worst case. The function PGrowthLB, which is an integral part of line 3, operates in $O(m \times |\mathcal{H}|)$ since each call to the LB_{OC} function is completed in $O(m)$. Handling the first filtering rule **R1** (lines $5-9$) has a time complexity of $O(n \times m \times |\mathcal{H}|)$ in the worst case. This is because the PGrowthLB function, which is invoked, runs in $O(m \times |\mathcal{H}|)$ and the size of $|\sigma^*|$ is bounded by n. The second filtering rule **R2** (lines $11-16$) is accomplished in $O(n^2 \times m)$. The inclusion test in line 12 is executed in $O(m)$. Given that $|\sigma^+| + |\sigma^-|$ is at most n, both **for** loops (on $|\sigma^+|$ and $|\sigma^-|$) require $n/2 \times n/2$ operations, resulting in an overall time complexity of $O(n^2 \times m)$. Now, considering that in most cases $|\mathcal{H}| > m$, the worst-case time complexity of Algorithm 1 is $O(n^2 \times |\mathcal{H}|)$.

□

1.9 Proof of Proposition 7 (Space Complexity)

Proof. Algorithm 1 relies on internal data structures, including $\sigma = \langle \sigma^+, \sigma^-, \sigma^* \rangle$, to maintain the partial assignment of $n = |\mathcal{I}|$ Boolean variables, requiring $O(n)$ space complexity. Additionally, it requires a set of diverse itemsets $\mathcal{H}$ mined so far. Since $|\mathcal{H}| \gg n$, the overall space complexity is $O(|\mathcal{H}|)$.

□

1.10 Proof of Proposition 8 (Domain Consistency)

Proof. The proposed monotonic LB_{OC} relaxation includes two filtering rules, **R1** and **R2**. These rules determine that any free item variable $x_i \in X$ that cannot be included in the current partial itemset σ^+ to form a diverse itemset is automatically filtered

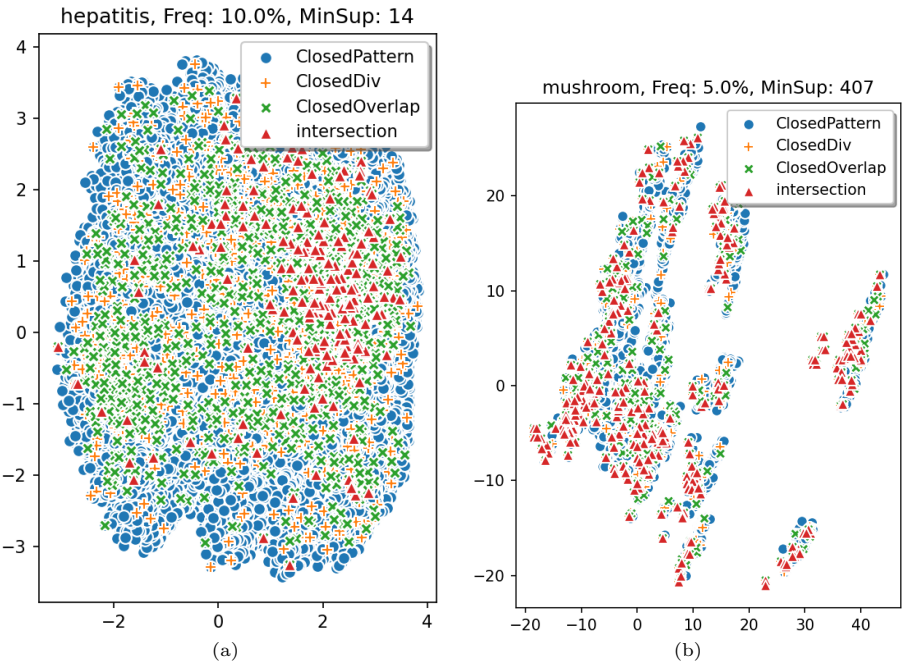


Fig. 3: Visual Qualitative Comparison of Diversified Closed Pattern Extractions in Principal Component Space

out, as explained in Proposition 7. Through construction, if any Boolean variable x_i such that $d_i \in D(x_i)$ does not satisfy the overlap constraint, it is pruned from $D(x_i)$. As a result, Algorithm 1 effectively ensures Domain Consistency for the set X.

□

2 Additional Experimental Results

2.1 Comparative Exploration of Extracted Pattern Landscapes through Principal Component Analysis (PCA)

Understanding the landscapes of extracted patterns across different mining approaches provides valuable insights into their relative performances. By visualizing patterns in a reduced principal component space, we can assess key qualities such as diversity, redundancy, and coverage of the extracted patterns. Comparative landscape analyses reveal critical visual indications of the strengths and weaknesses of pattern mining methods regarding the discrepancy, overlap, and distribution of the discovered patterns.

To this end, we perform a comparative analysis between `ClosedDiversity` and `ClosedOverlap`, examining the landscapes of their respective extracted pattern sets. The aim is to evaluate diversity and redundancy levels based on how different pattern extraction methods populate the pattern search space. This qualitative landscape assessment complements quantitative metrics, offering a holistic view of comparative

pattern quality. The scatter and distribution of patterns in the principal component space enable us to judge the relative coverage and variability in the extracted pattern sets for each approach.

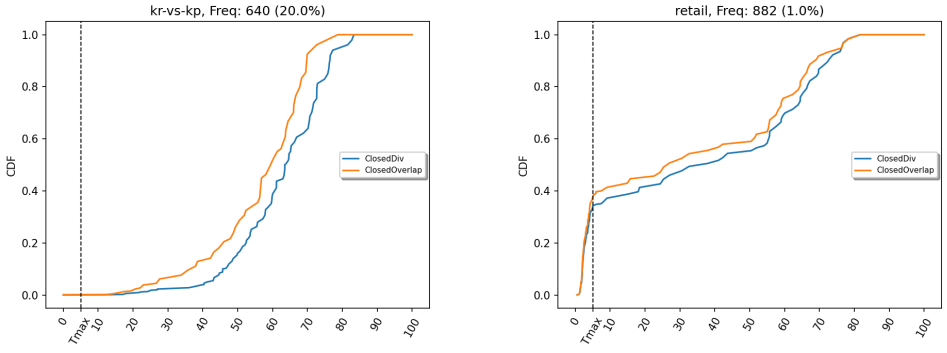
We apply Principal Component Analysis (PCA) using the pattern covers (boolean transaction sets) as features, treating each cover as a row in the input matrix. This approach captures the underlying variability and structure in the extracted pattern sets, enabling meaningful visualization. The results of the PCA are depicted in Figure 3, where we compare the pattern sets extracted by `ClosedDiversity` (orange pluses) and `ClosedOverlap` (green crosses).

The first two principal components explain a similar proportion of total variability for both methods, with `ClosedDiversity` accounting for approximately 60% and `ClosedOverlap` for 63%. However, the visual distribution in the PCA space reveals stark differences between the two approaches. The patterns extracted by `ClosedOverlap` (green crosses) are significantly more dispersed across the space, indicating greater diversity in the extracted patterns. In contrast, the orange pluses (`ClosedDiversity`) exhibit clustering, suggesting areas of redundancy where similar patterns aggregate. This clustering reflects lower diversity and higher overlap among the patterns extracted by `ClosedDiversity`.

Additionally, we observe red triangles representing patterns common to both `ClosedOverlap` and `ClosedDiversity`, as well as blue circles corresponding to all possible pattern candidates (the `ClosedPattern` space). The green crosses (`ClosedOverlap`) are better distributed across the representation space, demonstrating a broader exploration of the search space. These visual observations align with quantitative metrics, affirming `ClosedOverlap`'s superior performance in extracting closed patterns that are both frequent and more diversified. The global constraint based on the overlap coefficient allows `ClosedOverlap` to effectively reduce redundancy while enhancing diversity.

Figure 3 presents PCA visualizations for two datasets: the HEPATITIS-10 dataset (left) and the MUSHROOM-5 dataset (right). In both cases, `ClosedOverlap` consistently generates pattern sets with better dispersion and reduced redundancy compared to `ClosedDiversity`. These findings underline `ClosedOverlap`'s ability to produce high-quality, diverse pattern sets, offering a clear advantage in pattern mining tasks.

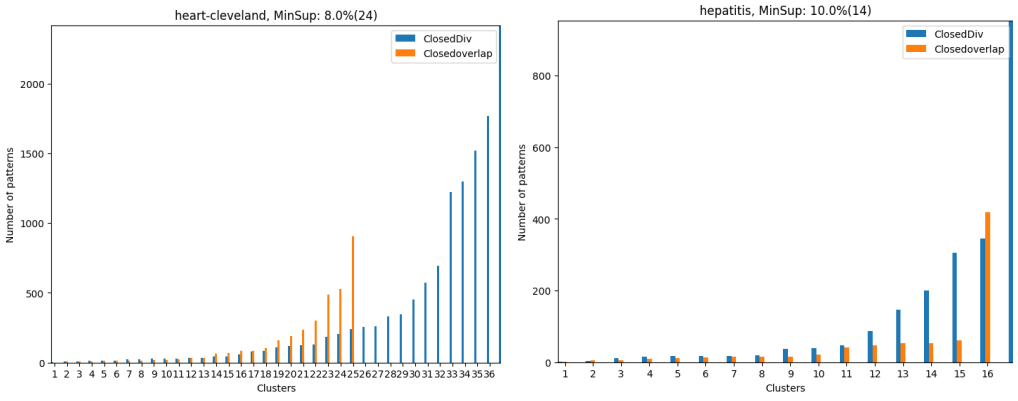
2.2 Assessing pattern set diversity through Cumulative Distribution Functions (CDF) (extension)



(a) (b)

Fig. 4: Comparing Pattern Diversity: `ClosedOverlap` vs. `ClosedDiversity` using Cumulative Density Functions (CDF)

2.3 Assessing Pattern Set Redundancy through Hierarchical Clustering (extension)



(a) (b)

Fig. 5: Comparison of number of undiversified(Similar) patterns in each clusters: `ClosedDiversity` vs `ClosedOverlap`

3 Typical CP Solver Search Mechanism

To solve a Constraint Satisfaction Problem (CSP), CP solvers typically employ **backtracking search** 1. At each step of the search tree, an unassigned variable is selected using user-defined heuristics and assigned a value from its domain. The solver backtracks if it encounters a constraint violation, i.e., when at least one domain becomes empty. One of the main techniques to speed up the search is **constraint propagation** using **filtering algorithms**. These algorithms enforce **local consistency** properties, such as **domain consistency**, which ensures that for every variable x_i in a constraint c, and for every value v in the domain of x_i, there exists an assignment that satisfies c with $x_i = v$.

Algorithm 1: ConstraintSearch(X, C, D)

Input : X: a set of decision variables; C: a set of constraints;
InOut: D: a set of variable domains;

```
1 D ← Propagate(D, C)
2 if there exists x_i ∈ X such that dom(x_i) is empty then
3     return failure
4 if there exists x_i ∈ X such that |dom(x_i)| > 1 then
5     Select x_i ∈ X such that |dom(x_i)| > 1
6     forall v ∈ dom(x_i) do
7         ConstraintSearch(X, C, D ∪ {x_i → v})
8 else
9     output solution D
```

Backtracking search

The algorithm 1, is designed to find a solution to a problem represented by a set of decision variables (X) and a set of constraints (C) on their domains. The algorithm takes an initial set of variable domains, represented by D, and applies a propagation step by invoking the function *Propagate(D, C)* to reduce the domains based on the given constraints (Line 1).

At each recursive step, the algorithm checks for specific conditions:

1. If there exists a decision variable x_i in X with an empty domain (i.e., no valid values left), the algorithm immediately returns "failure" (Lines 2-3). This indicates that the current assignment path does not lead to a valid solution.
2. If there exists a decision variable x_i in X with more than one value in its domain, the algorithm selects one of these variables (Lines 4-5).

   For each value v in the domain of the selected variable x_i, the algorithm performs a recursive call to *ConstraintSearch* with the updated domain $D \cup \{x_i \rightarrow v\}$, effectively considering the assignment of x_i to v (Lines 6-7). This recursive process explores different possible assignments and continues the search.

3. If all decision variables have only one value in their domains (i.e., no more choices left), the algorithm reaches a base case, and it outputs the solution D as a valid assignment to the decision variables (Lines 8-9). This indicates a successful path in the search space that satisfies all constraints.

The algorithm 1 uses backtracking to explore different paths in the search space, systematically reducing the possible solutions until it either finds a valid assignment or determines that no solution exists. The process terminates when all decision variables have single-value domains, and at that point, the algorithm has found a valid solution to the given problem.

3.1 Example of a CSP model

Formally, a CSP model, denoted by $\langle X, D, \mathcal{C} \rangle$, consists of a set X of n variables, a domain D mapping each variable $x_i \in X$ to a finite set of values D_i, and a set of constraints $\mathcal{C}$. An assignment σ is a mapping from variables in X to values in their domains: $\forall x_i \in X, \sigma(x_i) \in D_i$. A constraint $c \in \mathcal{C}$ is a subset of the cartesian product of the domains of the variables involved in c, denoted as $X(c)$. The objective is to find an assignment that fulfills all the specified constraints. In Constraint Programming, each constraint is considered as a sub-problem, so a problem can also be viewed as a conjunction of sub problems Régin (2011).

Example 2. *Consider the following CSP:*

$$
\begin{aligned}
X &= \{x_1, x_2, x_3\} \\
D &= \{D_1, D_2, D_3\} \ s.t., D_1 = \{1, 2\}, D_2 = \{0, 1, 2, 3\}, D_3 = \{2, 3\} \\
\mathcal{C} &= \{C_1(x_1, x_2), C_2(x_1, x_2, x_3), C_3(x_1, x_2)\} \ where, \\
&\quad C_1(x_1, x_2) : x_1 > x_2 \\
&\quad C_2(x_1, x_2, x_3) : x_1 + x_2 = x_3 \\
&\quad C_3(x_1, x_2) : x_1 \neq x_2
\end{aligned}
$$

Here, the current CSP admits two solutions: $(x_1 = 2, x_2 = 0, x_3 = 2)$ *and* $(x_1 = 2, x_2 = 1, x_3 = 3)$. *If we reconsider our example 2 by adding this constraint, denoted as* $C_4(x_1, x_2, x_3) =$ `all-different`(x_1, x_2, x_3), *we get the unique solution* $(x_1 = 2, x_2 = 1, x_3 = 3)$.

4 Global constraints-based model

Although the declarative advantage of the previous CP model, this encoding has a major drawback, as it requires the use of $(m = |\mathcal{T}|)$ reified constraints to represent the entire database. This limitation imposes restrictions on the size of databases that can be effectively handled. Interestingly, Lazaar *et al.* Lazaar et al. (2016) have introduced the `ClosedPattern` global constraint, which offers an efficient encoding method for both the minimal frequency and closedness constraints. This global constraint stands out as it eliminates the need for reified constraints or additional variables.

Definition 1 (`ClosedPattern` global constraint)**.** *`ClosedPattern`$_{\mathcal{D}}(X, \theta)$ holds iff there exists an assignment $\sigma = \langle d_1, ..., d_n \rangle$ of variables X s.t. $Freq_{\mathcal{D}}(\sigma^+) \geq \theta$ and $Clos_{freq}(\sigma^+)$.*

The `ClosedPattern` constraint encodes efficiently the minimal frequency and closeness constraints while ensuring domain consistency in polynomial time Lazaar et al. (2016).

Example 3. *Let's examine the transaction database in Table 1 with a minimum support threshold of $\theta = 2$. Suppose we have a pattern $X = \langle x_1, \ldots, x_5 \rangle$ with Boolean variables, where $D(x_i) = \{0, 1\}, i = 1..5$. In this case, consider the closed pattern ACD, represented by $X = \langle 10110 \rangle$, where $\sigma^+ = \{A, C, D\}$ and $\sigma^- = \{B, E\}$. We can verify that `ClosedPattern`$_{\mathcal{D}}(X, 2)$ holds because the pattern $\{A, C, D\}$ occurs at least 2 times in the transaction database ($Freq_{\mathcal{D}}(\{A, C, D\}) \geq 2$), and it satisfies the closedness constraint ($Clos_{freq}(\{A, C, D\})$).*

The `ClosedPattern` constraint, initially introduced by Lazaar *et al.* Lazaar et al. (2016), was further extended by Hien *et al.* Hien et al. (2021) to create a novel global constraint specifically tailored for mining closed and diverse itemsets. This new constraint, known as `ClosedDiversity`, leverages the filtering rules inherent to this extended constraint. Besides, the `ClosedPattern` constraint gave rise to two other effective constraints, each of which deals with a unique aspect: a constraint on frequency (`CoverSize`) and a specific constraint on closure (`CoverClosure`).

4.1 `CoverSize`

Definition 2. *Using bitwise operations, `CoverSize` can be expressed as:*

$$CoverSize([X_1, \ldots, X_n], \mathcal{D}, c) \Leftrightarrow c = size\left(\bigwedge_{i=1}^{n} \mathcal{V}_{\mathcal{D}}(X_i)\right) \tag{1}$$

where $\bigwedge$ signifies the bitwise AND operation.

Boolean variables X represent the selected items in the pattern. The database $\mathcal{D}$ is represented using vertical dense bitvectors, each corresponding to an item's occurrence in transactions. The integer variable c captures the number of transactions (bits) where all selected items are present. The constraint computes the intersection of the selected items bitvectors, and the count of set bits in this intersection is assigned to c. Bitwise AND operations are utilized to efficiently compute the intersection for dense bitvector representations.

5 Calculation of Joint Entropy

This section provides a clear step-by-step explanation of how the joint entropy is calculated from the binary matrix.

Initial History Matrix ($\mathcal{H}$)

$\{A, B, C, D\}$	$\{A, E\}$	$\{B, E\}$
0	0	1
0	0	0
0	0	0
1	1	1
1	0	0
0	0	0
0	1	0
0	1	0

Step 1: Identify Unique Bitcodes

Each row in the matrix represents a transaction as a combination of 1s (presence) and 0s (absence). The unique bitcodes in this matrix are:

$$\{001, 000, 111, 100, 010\}$$

Step 2: Count Occurrences of Each Bitcode

The frequency of each bitcode is:

$$\text{Occurrences:} \quad 001\text{: 1, } 000\text{: 3, } 111\text{: 1, } 100\text{: 1, } 010\text{: 2.}$$

Step 3: Calculate Probabilities

The probabilities are computed by dividing the occurrences of each bitcode by the total number of rows (transactions), $n = 8$:

$$P(001) = \frac{1}{8},\ P(000) = \frac{3}{8},\ P(111) = \frac{1}{8},\ P(100) = \frac{1}{8},\ P(010) = \frac{2}{8} = \frac{1}{4}.$$

Step 4: Compute Joint Entropy

The joint entropy H is:

$$\begin{aligned} H &= -\Bigg(P(001)\log_2 P(001) + P(000)\log_2 P(000) + P(111)\log_2 P(111) \\ &\quad + P(100)\log_2 P(100) + P(010)\log_2 P(010) \Bigg) \\ &= -\left(\frac{1}{8}\log_2 \frac{1}{8} + \frac{3}{8}\log_2 \frac{3}{8} + \frac{1}{8}\log_2 \frac{1}{8} + \frac{1}{8}\log_2 \frac{1}{8} + \frac{2}{8}\log_2 \frac{2}{8} \right). \\ &= \mathbf{2.1556\ bits} \end{aligned}$$

This value reflects the diversity and distribution in the history.

Note that the joint entropy is always between 0 and k, which is 3 in our example. If $H = k$, that means we have a **uniform distribution** (ideal case) and all patterns are independent.

References

Hien A, Loudni S, Aribi N, et al. (2021) A relaxation-based approach for mining diverse closed patterns. In: Hutter F, Kersting K, Lijffijt J, et al. (eds) Machine Learning and Knowledge Discovery in Databases. Springer International Publishing, Cham, pp 36–54

Lazaar N, Lebbah Y, Loudni S, et al. (2016) A global constraint for closed frequent pattern mining. In: Principles and Practice of Constraint Programming - 22nd International Conference, CP 2016, Toulouse, France, September 5-9, 2016, Proceedings, pp 333–349, https://doi.org/10.1007/978-3-319-44953-1_22, URL https://doi.org/10.1007/978-3-319-44953-1_22

Régin JC (2011) Global Constraints: A Survey, Springer New York, New York, NY, pp 63–134. https://doi.org/10.1007/978-1-4419-1644-0_3