Compressing and Querying Skypattern Cubes

Willy Ugarte, Samir Loudni, Patrice Boizumault, Bruno Crémilleux, Alexandre Termier

Compressing and Querying Skypattern Cubes

Willy Ugarte[1], Samir Loudni[2], Patrice Boizumault[2], Bruno Crémilleux[2], and Alexandre Termier[3]

[1] Peruvian University of Applied Sciences, Peru
willyugarte@gmail.com
[2] Normandie Univ., UNICAEN, ENSICAEN, CNRS – UMR GREYC, France
{name}.{last-name}@unicaen.fr
[3] Univ. Rennes, Inria, CNRS, IRISA, France
alexandre.termier@irisa.fr

Abstract. Skypatterns are important since they enable to take into account user preference through Pareto-dominance. Given a set of measures, a skypattern query finds the patterns that are not dominated by others. In practice, different users may be interested in different measures, and issue queries on any subset of measures (a.k.a subspace). This issue was recently addressed by introducing the concept of skypattern cubes. However, such a structure presents high redundancy and is not well adapted for updating operations like adding or removing measures, due to the high costs of subspace computations in retrieving skypatterns. In this paper, we propose a new structure called Compressed Skypattern Cube (abbreviated CSKYC), which concisely represents a skypattern cube, and gives an efficient algorithm to compute it. We thoroughly explore its properties and provide an efficient query processing algorithm. Experimental results show that our proposal allows to construct and to query a CSKYC very efficiently.

Keywords: skypatterns; pareto-dominance relation; skypattern cubes

1 Introduction

The notion of skyline queries [2] has been quite recently integrated into the pattern discovery paradigm to mine skyline patterns (henceforth called *skypatterns*) [11, 15]. Given a set of measures, skypatterns are based on a Pareto-dominance relation, which means that no measure can be improved without degrading the others. As an example, a user may prefer patterns with a high frequency, large size and a high confidence. Then a pattern $\mathtt{x_i}$ dominates another pattern $\mathtt{x_j}$ if $\mathtt{freq}(\mathtt{x_j}) \geq \mathtt{freq}(\mathtt{x_i})$, $\mathtt{size}(\mathtt{x_j}) \geq \mathtt{size}(\mathtt{x_i})$, $\mathtt{conf}(\mathtt{x_j}) \geq \mathtt{conf}(\mathtt{x_i})$ where at least one strict inequality holds. The skypattern set contains the patterns that are not dominated by any other pattern. Skypatterns are highly interesting since they do not require thresholds for the measures and the dominance relation gives them global interestingness with a semantics easily understood by the user.

In practice, users do not know the exact role of each measure and it is difficult to select beforehand the most appropriate subset of measures. Users would like to keep all potentially useful measures, look at what happens on skypattern sets when removing

or adding a measure, thus evaluating the impact of measures, and then converge to a convenient skypattern set.

This issue has been first addressed with the notion of a Skypattern Cube [13], which is the lattice of all possible subsets of measures associated with their skypattern sets. More formally, given a set M of n measures, the 2^n-1 possible non-empty skypattern subsets should be precomputed to efficiently handle various queries of users. By comparing two neighboring nodes (differentiated by only one measure), users can observe new skypatterns and the ones which disappear, greatly helping to better understand the role of the measures. To sum up, the cube is a structure that enables to discover the most interesting skypattern sets. The skypattern cube has been exploited in various domains such as bioinformatics [10] and mutagenicity [13]. However there are 2^n-1 possible non-empty skypattern sets with high redundancy coming from derivations of skypatterns among subspaces of the cube [13].

In this paper, we propose a new structure called the *Compressed Skypattern Cube* (denoted CSKYC). Each subspace stores skypatterns (called *proper skypatterns*) that do not appear in its descendant ones and the compressed skypattern cube contains only non-empty subspaces. Compared to the original skypattern cube [13], the CSKYC has fewer duplicates among subspaces, and does not need to store all of them. Moreover the cube includes unbalanced skypatterns. For instance, let M=$\{\texttt{freq}, \texttt{size}\}$ and three patterns $\texttt{x}_\texttt{i}$, $\texttt{x}_\texttt{j}$ and $\texttt{x}_\texttt{k}$ such that $\texttt{freq}(\texttt{x}_\texttt{i})$=10, $\texttt{size}(\texttt{x}_\texttt{i})$=1, $\texttt{freq}(\texttt{x}_\texttt{j})$=2, $\texttt{size}(\texttt{x}_\texttt{j})$=8, $\texttt{freq}(\texttt{x}_\texttt{k})$=4 and $\texttt{size}(\texttt{x}_\texttt{k})$=5. Clearly, $\texttt{x}_\texttt{i}$ (resp. $\texttt{x}_\texttt{j}$) is a skypattern for $\{\texttt{freq}\}$ (resp. $\{\texttt{size}\}$), thus $\texttt{x}_\texttt{i}$ (resp. $\texttt{x}_\texttt{j}$) will be instantly a skypattern for M being derived from $\{\texttt{freq}\}$ (resp. $\{\texttt{size}\}$). However, $\texttt{x}_\texttt{k}$ is also a skypattern for M, being more balanced over measures than $\texttt{x}_\texttt{i}$ (resp. $\texttt{x}_\texttt{j}$) which only has an extreme value for $\{\texttt{freq}\}$ (resp. $\{\texttt{size}\}$). Proper skypatterns are often well-balanced skypatterns.

Contributions overview. We thoroughly explore interesting properties of the compressed skypattern cube and provide an efficient query processing algorithm. Our contributions can be summarized as follows: (i) we provide the summarization structure CSKYC which concisely represents the whole skypattern cube and preserves its essential information. (ii) We propose a bottom-up approach to construct the CSKYC. (iii) We show how this structure can be used efficiently for query processing. Finally, (iv) we present an extensive set of experiments showing the advantages of our proposals.

Paper organization. The rest of this paper is organized as follows. Section 2 recalls the definitions of the notions used in this paper. Section 3 first introduces the CSKYC, provides algorithms to build it, and shows how the CSKYC can handle various skypattern queries. Section 4 is devoted to related works. Finally, Section 5 shows our experimental results and Section 6 concludes.

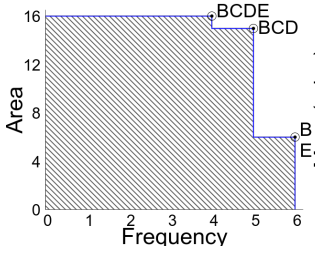
2 Preliminaries

Let $\mathcal{I}$ be a set of distinct literals called *items*. A pattern (or itemset) is a non-empty subset of $\mathcal{I}$. The language of patterns corresponds to $\mathcal{L}_\mathcal{I} = 2^\mathcal{I}\backslash\emptyset$. A transactional dataset $\mathcal{T}$ is a multiset of patterns in $\mathcal{L}_\mathcal{I}$. The traditional example is a supermarket database in which, for each transaction t_i, every item in a transaction is a product bought by the customer i. Tab. 1 summarizes the different notations used throughout the paper.

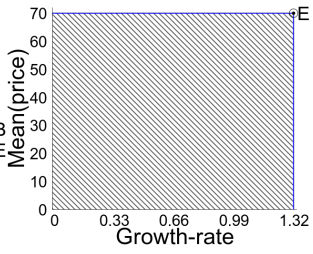
Trans.		Items					
t_1	c_1	A				E	F
t_2	c_1		B	C	D	E	
t_3	c_1		B	C	D	E	F
t_4	c_2	A	B	C	D	E	
t_5	c_2		B	C	D		
t_6	c_2		B			E	F
t_7	c_2	A	B	C	D	E	F

Item	A	B	C	D	E	F
Price	30	40	10	40	70	55

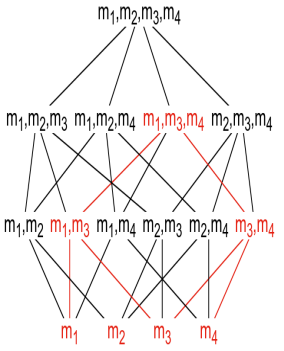
(a) Transactional dataset $\mathcal{T}$.



(b) Skypatterns for $\{\mathsf{m}_1, \mathsf{m}_3\}$.



(c) Skypatterns for $\{\mathsf{m}_2, \mathsf{m}_4\}$.



(d) Lattice associated to $\mathcal{M}$.

Subset of $\mathcal{M}$	Skypatterns
$\{\mathsf{m}_1, \mathsf{m}_2, \mathsf{m}_3, \mathsf{m}_4\}$	BCD, BCDE, BDE, BE, E, EF
$\{\mathsf{m}_1, \mathsf{m}_2, \mathsf{m}_3\}$	BCD, BCDE, E
$\{\mathsf{m}_1, \mathsf{m}_2, \mathsf{m}_4\}$	E
$\{\mathsf{m}_1, \mathsf{m}_3, \mathsf{m}_4\}$	BCD, BCDE, BDE, BE, E, EF
$\{\mathsf{m}_2, \mathsf{m}_3, \mathsf{m}_4\}$	BCDE, BDE, E, EF
$\{\mathsf{m}_1, \mathsf{m}_2\}$	E
$\{\mathsf{m}_1, \mathsf{m}_3\}$	BCD, BCDE, B, E
$\{\mathsf{m}_1, \mathsf{m}_4\}$	E
$\{\mathsf{m}_2, \mathsf{m}_3\}$	BCDE
$\{\mathsf{m}_2, \mathsf{m}_4\}$	E
$\{\mathsf{m}_3, \mathsf{m}_4\}$	BCDE, BDE, E, EF
$\{\mathsf{m}_1\}$	B, E
$\{\mathsf{m}_2\}$	AEF, BCDE, BCDEF, BDE, E, BCDF, BDEF, BDF, AF, EF, F
$\{\mathsf{m}_3\}$	BCDE
$\{\mathsf{m}_4\}$	E

(e) $SKYC$ for $\mathcal{M}$.

Subset of $\mathcal{M}$	Proper Skypatterns
$\{\mathsf{m}_1, \mathsf{m}_3, \mathsf{m}_4\}$	BE
$\{\mathsf{m}_1, \mathsf{m}_3\}$	BCD
$\{\mathsf{m}_3, \mathsf{m}_4\}$	BDE, EF
$\{\mathsf{m}_1\}$	B, E
$\{\mathsf{m}_2\}$	AEF, AF, BDE, BCDEF, BCDF, BCDE, BDEF, BDF, E, EF, F
$\{\mathsf{m}_3\}$	BCDE
$\{\mathsf{m}_4\}$	E

(f) $CSKYC$ for $\mathcal{M}$.

Fig. 1: Running example: $\mathcal{M} = \{\ \mathsf{m}_1{:}\mathtt{freq}(\mathtt{x}), \mathsf{m}_2{:}\mathtt{gr}_1(\mathtt{x}), \mathsf{m}_3{:}\mathtt{area}(\mathtt{x}), \mathsf{m}_4{:}\mathtt{mean}(\mathtt{x}.\mathrm{price})\ \}$.

Example 1. Fig. 1a depicts a transactional dataset $\mathcal{T}$ where items in a transaction t_i are denoted $A, \ldots, F$. It serves as example throughout the paper. An attribute (*price*) is associated to each item. For instance, the *Price* of A is \$30. The dataset is partitioned into two classes, class c_1 for clients with loyalty programs and class c_2 for other clients.

Constraint-based pattern mining aims at extracting all patterns $\mathtt{x} \in \mathcal{L}_\mathcal{I}$ satisfying a query $q(\mathtt{x})$ which is usually called *theory* [7]: $Th(\mathcal{L}_\mathcal{I}, q) = \{\ \mathtt{x} \in \mathcal{L}_\mathcal{I}\ \mid\ q(\mathtt{x})\ \ is\ \ true\ \}$. A common example is the minimal frequency constraint ($\mathtt{freq}(\mathtt{x}) \geq \theta$) which provides patterns having a number of occurrences exceeding a given minimal threshold θ. Many other measures for patterns can be considered such as:

- $\mathtt{size}(\mathtt{x}) = |\mathtt{x}|$ is the number of items that x contains.
- $\mathtt{gr}_1(\mathtt{x}) = \frac{(|\mathcal{T}| - |\mathcal{T}_1|) \times \mathtt{freq}_1(\mathtt{x})}{|\mathcal{T}_1| \times (\mathtt{freq}(\mathtt{x}) - \mathtt{freq}_1(\mathtt{x}))}$ where $\mathcal{T}_1$ is a sub-dataset (i.e a class partition) on $\mathcal{T}$.
- $\mathtt{min}(\mathtt{x}.att) = \min\limits_{i \in \mathtt{x}} \{i.att\}$ (resp. $\mathtt{max}(\mathtt{x}.att) = \max\limits_{i \in \mathtt{x}} \{i.att\}$) is the lowest (resp. highest) among item values of x for attribute att.
- $\mathtt{mean}(\mathtt{x}.att) = (\mathtt{min}(\mathtt{x}.att) + \mathtt{max}(\mathtt{x}.att))/2$.

Example 2. $\mathtt{freq}(BC) = 5, \mathtt{mean}(BCD.price) = 25, \ldots$

Skypatterns allow to express a user-preference according to a dominance relation [11].

Table 1: Notations.

Symbol	Definition
$\mathcal{T}$	Transactional dataset.
$\mathcal{I}$	Set of items.
$\mathcal{L}_\mathcal{I} = 2^\mathcal{I} \setminus \emptyset$	Language of patterns.
$\mathcal{M}$	Set of measures.
$\mathrm{U} \subseteq \mathcal{M}$	Subspace U (i.e. subset of measures of $\mathcal{M}$).
$Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})$	Skypatterns set over $\mathcal{L}_\mathcal{I}$ for U.
$\boldsymbol{P}$-$Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})$	Proper Skypattern set over $\mathcal{L}_\mathcal{I}$ for U.
$\boldsymbol{\ell}$-$Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})$	Large Skypattern set over $\mathcal{L}_\mathcal{I}$ for U.
$Desc(\mathcal{L}_\mathcal{I}, \mathrm{U})$	Union of all proper skypattern sets for all descendant subspaces V $\subset$ U.
$SKYC(\mathcal{L}_\mathcal{I}, \mathcal{M})$	Skypattern Cube over $\mathcal{L}_\mathcal{I}$ for $\mathcal{M}$.
$CSKYC(\mathcal{L}_\mathcal{I}, \mathcal{M})$	Compressed Skypattern Cube over $\mathcal{L}_\mathcal{I}$ for $\mathcal{M}$.
$P \subseteq \mathcal{L}_\mathcal{I}$	Set of patterns.
$\mathcal{O}(P)$	Set of data points associated to P.
$Skyline(\mathcal{O}(P), \mathrm{U})$	Set of skyline points on $\mathcal{O}(P)$ for U.

Definition 1 (Pareto-dominance). *A pattern* $\mathtt{x}$ *dominates another pattern* $\mathtt{y}$ *w.r.t a measure subset (a.k.a subspace) U, noted by* $\mathtt{x} \succ_U \mathtt{y}$*, iff* $\forall \mathsf{m_i} \in U, \mathsf{m_i}(\mathtt{x}) \geq \mathsf{m_i}(\mathtt{y})$ *and* $\exists \mathsf{m_j} \in U, \mathsf{m_j}(\mathtt{x}) > \mathsf{m_j}(\mathtt{y})$.

Example 3. For $\mathrm{U} = \{\mathsf{m_1}{:}\mathtt{freq}(\mathtt{x}), \mathsf{m_3}{:}\mathtt{area}(\mathtt{x})\}$, pattern BCD dominates pattern BC since $\mathtt{freq}(BCD) = \mathtt{freq}(BC) = 5$ and $\mathtt{area}(BCD) > \mathtt{area}(BC)$.

The Skypattern Operator [11] extracts the skypattern set w.r.t a subspace U.

Definition 2 (Skypattern Operator). *A pattern is a skypattern w.r.t a subspace U iff it is not dominated by any other pattern w.r.t U. The Skypattern Operator returns all the skypatterns w.r.t U:* $Sky(\mathcal{L}_\mathcal{I}, U) = \{\ \mathtt{x} \in \mathcal{L}_\mathcal{I} \ \mid \ \nexists\, \mathtt{y} \in \mathcal{L}_\mathcal{I}, \mathtt{y} \succ_U \mathtt{x}\ \}$.

Example 4. Consider the dataset of Fig. 1a, users may ask for the skypatterns for every combination of the measures $\{\mathsf{m_1}, \mathsf{m_2}, \mathsf{m_3}, \mathsf{m_4}\}$. Figs. 1b and 1c depict skypatterns for $\{\mathsf{m_1}, \mathsf{m_3}\}$ and $\{\mathsf{m_2}, \mathsf{m_4}\}$ respectively.

As said above, users may query multiple skypattern sets for different subspaces. Furthermore, for a space $\mathcal{M}$ there are $2^{|\mathcal{M}|}-1$ different skypattern sets. The Skypattern Cube [13] retrieves all skypattern sets for any subspace.

Definition 3 (Skypattern Cube). *Given a set of measures* $\mathcal{M}$*, the Skypattern Cube of* $\mathcal{M}$ *is defined as* $SKYC(\mathcal{L}_\mathcal{I}, \mathcal{M}) = \{\ (U, Sky(\mathcal{L}_\mathcal{I}, U)) \ \mid \ U \subseteq \mathcal{M}, U \neq \emptyset\ \}$

Example 5. Fig. 1d depicts the lattice associated to $\mathcal{M}$ (power set of $\mathcal{M}$: $2^\mathcal{M} \setminus \emptyset$). Fig. 1e associates to each non-empty subset of $\mathcal{M}$ its skypattern set.

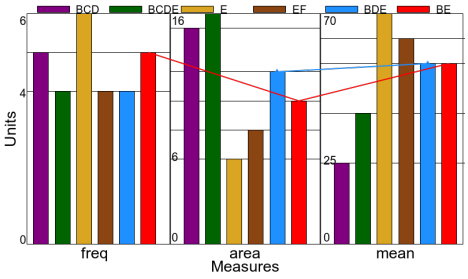
For computing the skypattern cube, [13] has proposed a bottom-up approach using two derivation rules that provide an easy way to automatically infer a large proportion of the skypatterns of a parent node from the skypattern sets of its child nodes without any dominance test (if k measures are associated to a parent node, its child nodes are the nodes defined by the $\binom{k}{k-1}$ subsets of $k-1$ measures).

$Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})$

Pattern	freq	area	mean(x.*price*)
BCD	5	15	25.00
BCDE	4	16	40.00
E	6	6	70.00
EF	4	8	62.50
BDE	4	12	55.00
BE	5	10	55.00

$Sky(\mathcal{L}_\mathcal{I}, \mathrm{U}')$

(a) Skypatterns for U and U′.



(b) Measure values of skypatterns.

Fig. 2: Example of (Proper) Skypatterns for $\mathrm{U} = \{\mathsf{m}_1, \mathsf{m}_3, \mathsf{m}_4\}$ and $\mathrm{U}' = \{\mathsf{m}_3, \mathsf{m}_4\}$.

3 Contributions

This section introduces the CSKYC (Compressed Skypattern Cubes) which concisely represents the entire skypattern cube. The main idea is, for every subspace, to only store its *proper skypatterns*. More precisely, a skypattern x for a subspace U is stored iff $\mathrm{x} \in Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})$ and there exists no $\mathrm{V} \subset \mathrm{U}$ s.t. $\mathrm{x} \in Sky(\mathcal{L}_\mathcal{I}, \mathrm{V})$. First, we introduce the CSKYC of a set of measures. Then, we propose a bottom-up approach for building such a CSKYC. Finally, we show how to efficiently query the whole skypattern set for U from the CSKYC.

3.1 The Compressed Skypattern Cube

Definition 4 (Proper Skypattern). *The set of proper skypatterns for a subspace U is the subset of skypatterns on U which are not skypatterns in any subset of U:*

$$\boldsymbol{P}\text{-}Sky(\mathcal{L}_\mathcal{I}, U) = \{\, \mathrm{x} \in Sky(\mathcal{L}_\mathcal{I}, U) \mid \nexists V \subset U, \mathrm{x} \in Sky(\mathcal{L}_\mathcal{I}, V) \,\}$$

Example 6. Consider $\mathrm{U} = \{\mathsf{m}_1, \mathsf{m}_3, \mathsf{m}_4\}$ and $\mathrm{U}' = \{\mathsf{m}_3, \mathsf{m}_4\}$. Tab. 2a shows the measure values for skypatterns for U and U′. Fig. 2b illustrates how proper skypattern BE (in red) (resp. BDE (in light-blue)) is more balanced than other skypatterns for U (resp. U′), having fewer extreme values than the other skypatterns.

Based on this notion, we define the compressed skypattern cube.

Definition 5 (Compressed Skypattern Cube). *Given a set of measures $\mathcal{M}$, the compressed skypattern cube of $\mathcal{M}$ is defined as*

$$CSKYC(\mathcal{L}_\mathcal{I}, \mathcal{M}) = \{\, (U, \boldsymbol{P}\text{-}Sky(\mathcal{L}_\mathcal{I}, U)) \mid U \subseteq \mathcal{M}, U \neq \emptyset, \boldsymbol{P}\text{-}Sky(\mathcal{L}_\mathcal{I}, U) \neq \emptyset \,\}$$

Example 7. For the dataset shown in Fig. 1a, the $CSKYC(\mathcal{L}_\mathcal{I}, \mathcal{M})$ is depicted in Fig. 1f and its sub-lattice (in red) in Fig. 1d. It contains only 6 non-empty subsets compared to 15 subsets in $SKYC(\mathcal{L}_\mathcal{I}, \mathcal{M})$. Clearly, the CSKYC is much more compact.

3.2 Computing the CSKYC

A first and naive way to get the CSKYC consists in first computing the skypattern cube, and then deriving the CSKYC by removing duplicates from their subspaces. Such an approach is inefficient as the number of subspaces to process is exponential. In this section, we provide a bottom-up algorithm (CSKYC−BUC) for building the CSKYC. Given a set of measures $\mathcal{M}$ of size d, the subspaces are organized into d levels, such that the subspaces of size i are in level i. We only keep *non-empty* subspaces (i.e. those containing proper skypatterns). All descendant skypatterns of a subspace are collected to form a large skypattern set ($\boldsymbol{\ell}$-Sky) which are then used as filters, and if no new skypattern is found, the subspace is discarded from the CSKYC.
Let us first give some preliminary definitions in order to compute the CSKYC.

Definition 6 (Indistinct/Incomparable Skypatterns). *Let* $\mathtt{x}, \mathtt{y}$ *be two skypatterns w.r.t a subspace U: (i)* $\mathtt{x}, \mathtt{y}$ *are indistinct, noted* $\mathtt{x} =_U \mathtt{y}$*, iff* $\forall \mathsf{m_i} \in U, \mathsf{m_i}(\mathtt{x}) = \mathsf{m_i}(\mathtt{y})$*; (ii)* $\mathtt{x}, \mathtt{y}$ *are incomparable, noted* $\mathtt{x} \prec\!\!\succ_U \mathtt{y}$*, iff* $\mathtt{x} \not\succ_U \mathtt{y}$, $\mathtt{y} \not\succ_U \mathtt{x}$ *and* $\mathtt{x} \neq_U \mathtt{y}$.

Incomparable skypatterns and indistinct ones for U constitute partitions of $Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})$.

Definition 7 (Indistinct Subspace (IS)). *A subspace U is an* ***Indistinct Subspace (IS)*** *iff all patterns in* $Sky(\mathcal{L}_\mathcal{I}, U)$ *are indistinct from each other.*

Example 8. Let $\mathrm{U} = \{\mathsf{m_1}\}$, and $\mathrm{V} = \{\mathsf{m_1}, \mathsf{m_3}\}$. B and E are indistinct w.r.t. U, while BCDE and BCD are incomparable w.r.t. V.

Lemma 1 states that skypatterns that are common to two different subspaces remain skypatterns in their union.

Lemma 1. $Sky(\mathcal{L}_\mathcal{I}, U) \cap Sky(\mathcal{L}_\mathcal{I}, V) \subseteq Sky(\mathcal{L}_\mathcal{I}, U \cup V)$

Proof (By contradiction). Assume that, for two subspaces U, V s.t. $\mathrm{W} = \mathrm{U} \cup \mathrm{V}$,
$\underbrace{\exists \mathtt{x} \in Sky(\mathcal{L}_\mathcal{I}, \mathrm{U}) \cap Sky(\mathcal{L}_\mathcal{I}, \mathrm{V})}_{(1)}$, but $\underbrace{\mathtt{x} \notin Sky(\mathcal{L}_\mathcal{I}, \mathrm{W})}_{(2)}$. From (1): $\underbrace{\mathtt{x} \in Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})}_{(3)}$ and
$\underbrace{\mathtt{x} \in Sky(\mathcal{L}_\mathcal{I}, \mathrm{V})}_{(4)}$. From (2): $\exists \mathtt{y} \in Sky(\mathcal{L}_\mathcal{I}, \mathrm{W}), \mathtt{y} \succ_\mathrm{W} \mathtt{x} \Rightarrow \underbrace{\forall \mathsf{m_i} \in \mathrm{W}, \mathsf{m_i}(\mathtt{y}) \geq \mathsf{m_i}(\mathtt{x})}_{(5)}$.

$$\left.\begin{array}{l}\text{From (3): } \mathtt{y} \not\succ_\mathrm{U} \mathtt{x} \Rightarrow \forall \mathsf{m_i} \in \mathrm{U}, \mathsf{m_i}(\mathtt{y}) \leq \mathsf{m_i}(\mathtt{x}). \text{ From (5): } \mathtt{x} =_\mathrm{U} \mathtt{y} \\ \text{From (4): } \mathtt{y} \not\succ_\mathrm{V} \mathtt{x} \Rightarrow \forall \mathsf{m_i} \in \mathrm{V}, \mathsf{m_i}(\mathtt{y}) \leq \mathsf{m_i}(\mathtt{x}). \text{ From (5): } \mathtt{x} =_\mathrm{V} \mathtt{y}\end{array}\right\} \mathtt{x} =_\mathrm{W} \mathtt{y}.$$

Thus, $\mathtt{x} \in Sky(\mathcal{L}_\mathcal{I}, \mathrm{W})$ leading to a contradiction.

Based on lemma 1, the following theorem enables us to characterize empty subspaces in the CSKYC, i.e. those without proper skypatterns.

Theorem 1 (Empty subspaces in CSKYC). *Given two subspaces U and V that are IS, if* $Sky(\mathcal{L}_\mathcal{I}, U) \cap Sky(\mathcal{L}_\mathcal{I}, V) \neq \emptyset$*, then* $U \cup V$ *is an IS and* $\boldsymbol{P}$-$Sky(\mathcal{L}_\mathcal{I}, U \cup V) = \emptyset$.

Proof (By contradiction). Let U, V two IS and $\mathrm{W} = \mathrm{U} \cup \mathrm{V}$.
- Assume that W is not an IS and $\exists \mathtt{x}, \mathtt{y} \in Sky(\mathcal{T}, \mathrm{U}) \cap Sky(\mathcal{T}, \mathrm{V})$. From Lemma 1: $\mathtt{x}, \mathtt{y} \in Sky(\mathcal{T}, \mathrm{W})$. Since W is not an IS, $\mathtt{x} \prec\!\!\succ_\mathrm{W} \mathtt{y}$. As $\mathtt{x}, \mathtt{y} \in Sky(\mathcal{T}, \mathrm{U})$, $\mathtt{x}, \mathtt{y} \in Sky(\mathcal{T}, \mathrm{V})$

and U and V are IS, thus, $\mathtt{x} =_{\mathrm{U}} \mathtt{y}$ and $\mathtt{x} =_{\mathrm{V}} \mathtt{y}$. Thus, $\mathtt{x} =_{\mathrm{W}} \mathtt{y}$ leading to a contradiction. - Assume that $\exists \mathtt{x} \in Sky(\mathcal{T}, \mathrm{U}) \cap Sky(\mathcal{T}, \mathrm{V})$ and $\exists \mathtt{y} \in \boldsymbol{P}\text{-}Sky(\mathcal{T}, \mathrm{W})$. From Lemma 1: $\mathtt{x} \in Sky(\mathcal{T}, \mathrm{W})$. Since W is an IS, $\mathtt{x} =_{\mathrm{W}} \mathtt{y}$. Thus, $\mathtt{x} =_{\mathrm{U}} \mathtt{y}$ and $\mathtt{x} =_{\mathrm{V}} \mathtt{y}$. So, $\mathtt{y} \in Sky(\mathcal{T}, \mathrm{U})$ and $\mathtt{y} \in Sky(\mathcal{T}, \mathrm{V})$. Thus, $\mathtt{y} \notin \boldsymbol{P}\text{-}Sky(\mathcal{T}, \mathrm{W})$ leading to a contradiction.

Example 9. In Fig. 1f, $\boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_1, \mathsf{m}_4\}) = \emptyset$ as $Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_1\}) \cap Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_4\}) = \{E\}$

The authors in [13] showed that incomparable skypatterns and some indistinct skypatterns of a child subspace remain also skypatterns in its parent subspace (they are referred to as derivable skypatterns). They also showed that a parent subspace can include non-derivable skypatterns (i.e., those that are not skypatterns in any of its child subspaces). Thus, one can collect the non-empty sets of descendants (which are proper skypatterns) of a subspace to form a large skypattern set and to use them as filters to detect a priori that no proper skypattern exist (see Corollary 1).

Definition 8 (Large Skypattern Set). *The Large Skypattern Set for a subspace U is the union of proper skypattern set of U with all proper skypattern sets of its descendant subspaces $V \subset U$: $\boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, U) = \boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, U) \cup Desc(\mathcal{L}_{\mathcal{I}}, U)$,*
where $Desc(\mathcal{L}_{\mathcal{I}}, U) = \bigcup_{V \subset U} \boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, V) = \bigcup_{V \subset U \wedge |V| = |U|-1} \boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, V)$

Example 10. For $\mathrm{U} = \{\mathsf{m}_1{:}\mathtt{freq}(\mathtt{x}), \mathsf{m}_3{:}\mathtt{area}(\mathtt{x}), \mathsf{m}_4{:}\mathtt{mean}(\mathtt{x.price})\}$:

$$\boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U}) = \boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U}) \cup \boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_1, \mathsf{m}_3\}) \cup \boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_1, \mathsf{m}_4\}) \cup \boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_3, \mathsf{m}_4\})$$

$$\overbrace{\boxed{B, E, EF, BCD, BE, BDE, BCDE}}$$

$$\overbrace{\boxed{\begin{array}{c}\boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_1, \mathsf{m}_3\}) \\ \cup \\ \boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_1\}) \\ \cup \\ \boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_3\})\end{array}}}^{\boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_1, \mathsf{m}_3\})} \quad \overbrace{\boxed{\begin{array}{c}\boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_1, \mathsf{m}_4\}) \\ \cup \\ \boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_1\}) \\ \cup \\ \boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_4\})\end{array}}}^{\boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_1, \mathsf{m}_4\})} \quad \overbrace{\boxed{\begin{array}{c}\boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_3, \mathsf{m}_4\}) \\ \cup \\ \boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_3\}) \\ \cup \\ \boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_4\})\end{array}}}^{\boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \{\mathsf{m}_3, \mathsf{m}_4\})}$$

Based on Definition 4, the proper skypatterns of any parent subspace can be computed thanks to the following corollary.

Corollary 1. $\boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, U) = Sky(\mathcal{L}_{\mathcal{I}}, U) \setminus Desc(\mathcal{L}_{\mathcal{I}}, U)$.

To compute $\boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U})$, we first retrieve its descendants (which are proper skypatterns), then we seek for skypatterns that are not in $Desc(\mathcal{L}_{\mathcal{I}}, \mathrm{U})$. Algorithm 1 gives the pseudo-code of our bottom-up approach. It starts by computing $\boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \mathsf{m}_{\mathsf{i}})$ for every $\mathsf{m}_{\mathsf{i}} \in \mathcal{M}$ (level 1) and then follows a level-wise strategy: from the lower level, each level of the lattice is constructed by applying Theorem 1 and, if needed, computing non-derivable skypatterns (cf. line 21). Two data structures, IS and $\boldsymbol{\ell}\text{-}Sky$ are also maintained during the construction process, storing for each subspace its large pattern set and its status. They allow an incremental computation of $\boldsymbol{\ell}\text{-}Sky$.

3.3 Querying $Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U})$ from $CSKYC(\mathcal{L}_{\mathcal{I}}, \mathcal{M})$

When a skypattern set for a given subspace U is queried, the CSKYC may not have a record for U; even if it does, the skypattern set that is stored for U is not complete. We propose a straightforward approach to query the complete skypattern set for U from $CSKYC(\mathcal{L}_{\mathcal{I}}, \mathcal{M})$.

Algorithm 1: CSKYC−BUC: Bottom-up approach for computing CSKYC.

Input: $\mathcal{T}$: a dataset, $\mathcal{M}$: a set of measures.
Output: The Compressed Skypattern Cube w.r.t $\mathcal{M}$.

CSKYC $\leftarrow \emptyset$; IS[ ] $\leftarrow \emptyset$; ℓ-Sky[ ] $\leftarrow \emptyset$; // Initialization step
foreach $\mathsf{m_i} \in \mathcal{M}$ **do**
  $\boldsymbol{P}$-$Sky \leftarrow$ CP+SKY$(\mathcal{L}_\mathcal{I}, \{\mathsf{m_i}\})$; // Compute $\boldsymbol{P}$-$Sky(\mathcal{L}_\mathcal{I}, \mathsf{m_i})$
  ℓ-$Sky[\{\mathsf{m_i}\}] \leftarrow \boldsymbol{P}$-$Sky$;
  CSKYC $\leftarrow$ CSKYC $\cup$ $\{(\{\mathsf{m_i}\}, \boldsymbol{P}\text{-}Sky)\}$;
  IS$[\{\mathsf{m_i}\}] \leftarrow true$;
for $i \leftarrow 2$ *to* $|\mathcal{M}|$ **do**
  foreach $U \subseteq \mathcal{M}$ *s.t.* $|U| = i$ **do**
    $\boldsymbol{P}$-$Sky \leftarrow$ ComputeProperSky(U); // Compute $\boldsymbol{P}$-$Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})$
    if $\boldsymbol{P}$-$Sky \neq \emptyset$ **then**
      CSKYC $\leftarrow$ CSKYC $\cup$ $\{(\mathrm{U}, \boldsymbol{P}\text{-}Sky)\}$;
return *CSKYC*;
Function ComputeProperSky(U)**:**
  children $\leftarrow$ { V $\subset$ U | $|\mathrm{V}| = |\mathrm{U}| - 1$ }; // Children of U
  childrenIS $\leftarrow$ { W $\in$ children | IS[W] $= true$ };
  IS[U] $\leftarrow false$;
  if $\exists V, W \in$ childrenIS *s.t.* $Sky[V] \cap Sky[W] \neq \emptyset$ **then** // Apply theorem 1
    $\boldsymbol{P}$-$Sky \leftarrow \emptyset$;
    IS[U] $\leftarrow true$;
  $Desc \leftarrow \bigcup_{\mathrm{V} \in children} \ell\text{-}Sky[\mathrm{V}]$; // Generate the filter skypatterns
  if $\neg$IS$[U]$ **then**
    $\boldsymbol{P}$-$Sky \leftarrow$ CP+SKY$(\mathcal{L}_\mathcal{I} \setminus Desc, \mathrm{U})$; // Apply corollary 1
  ℓ-Sky[U] $\leftarrow \boldsymbol{P}$-$Sky \cup Desc$; // Update ℓ-Sky for U
  return $\boldsymbol{P}$-Sky

Our approach is based on the fact that $Sky(\mathcal{L}_\mathcal{I}, \mathrm{U}) \subseteq \ell\text{-}Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})$ and proceeds in two steps (see Algorithm 2): first, approximating $Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})$ by ℓ-$Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})$, and then, performing domination tests to filter dominated patterns.

(i) *Approximating* $Sky(\mathcal{L}_\mathcal{I}, U)$.
Based on Definition 8, we have that: $\forall$ U $\subseteq \mathcal{M}$, $Sky(\mathcal{L}_\mathcal{I}, \mathrm{U}) \subseteq \ell\text{-}Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})$. The proof is straightforward: $\forall$ $\mathtt{x} \in Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})$, either $\mathtt{x} \in \boldsymbol{P}\text{-}Sky(\mathcal{L}_\mathcal{I}, \mathrm{U})$, or $\exists$V $\subset$ U s.t. $\mathtt{x} \in \boldsymbol{P}\text{-}Sky(\mathcal{L}_\mathcal{I}, \mathrm{V})$.

(ii) *Filtering dominated skypatterns*.
To remove dominated skypatterns, we convert the problem into skyline mining operation in |U| dimensions to process it more efficiently. Let f be a mapping function from a set of patterns $P \subseteq \mathcal{L}_\mathcal{I}$ to $\mathbb{R}^n$ that associates, to each pattern $\mathtt{x}_i \in P$, a data point $f(\mathtt{x}_i) \in \mathbb{R}^n$ with coordinates $(\mathsf{m_1}(\mathtt{x}_i) = v_{i,1}, \ldots, \mathsf{m_n}(\mathtt{x}_i) = v_{i,n})$. Let us note by $\mathcal{O}(P) = \{ f(\mathtt{x}) \mid \mathtt{x} \in P \}$ the set of data points associated to P (see Tab. 2) and $Skyline(\mathcal{O}(P), \mathrm{U})$ be the set of skyline points of $\mathcal{O}(P)$ w.r.t. U. Thus, $\forall$U $\subseteq \mathcal{M}$,

Table 2: The multidimensional view for a set of patterns $P \subseteq \mathcal{L}_{\mathcal{I}}$ w.r.t. a subspace U ($|\mathrm{U}| = \mathsf{n}$).

	Pattern	m_1	m_2	$\ldots$	m_n
$P \subseteq \mathcal{L}_{\mathcal{I}}$	$\mathtt{x}_1$	$v_{1,1}$	$v_{1,2}$	$\ldots$	$v_{1,\mathsf{n}}$
	$\mathtt{x}_2$	$v_{2,1}$	$v_{2,2}$	$\ldots$	$v_{2,\mathsf{n}}$
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	$\mathtt{x}_{p-1}$	$v_{p-1,1}$	$v_{p-1,2}$	$\ldots$	$v_{p-1,\mathsf{n}}$
	$\mathtt{x}_p$	$v_{p,1}$	$v_{p,2}$	$\ldots$	$v_{p,\mathsf{n}}$

Algorithm 2: Querying $Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U})$ from $CSKYC(\mathcal{L}_{\mathcal{I}}, \mathcal{M})$

Input: U: a subspace and $CSKYC$: the compressed skyppatern cube w.r.t. $\mathcal{M}$.
Output: $Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U})$

1 $\boldsymbol{\ell}\text{-}Sky \leftarrow \bigcup_{\mathrm{V} \subseteq \mathrm{U}} \boldsymbol{P}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{V})$; `// Approximating` $Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U})$

2 $Sky \leftarrow \mathtt{BNL}(\mathcal{O}(\boldsymbol{\ell}\text{-}Sky), \mathrm{U})$; `// Filtering dominated skypatterns`

3 **return** Sky, $\boldsymbol{\ell}\text{-}Sky$

$Sky(P, \mathrm{U}) = Skyline(\mathcal{O}(P), \mathrm{U})$. So, applying the skyline operator on $\mathcal{O}(P)$ provides the skypattern set.

Theorem 2. $Sky(\mathcal{L}_{\mathcal{I}}, U) = Skyline(\mathcal{O}(\boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, U)), U)$.

Proof. Given a subspace U, we prove the two implications:
($\Rightarrow$) Assume that $\exists \mathtt{x} \in Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U})$: $\forall \mathtt{y} \in \mathcal{L}_{\mathcal{I}}, \mathtt{y} \not\succ_{\mathrm{U}} \mathtt{x}$. So, $\forall \mathtt{y} \in \boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U}), \mathtt{y} \not\succ_{\mathrm{U}} \mathtt{x}$ therefore $\mathtt{x} \in Sky(\boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U}), \mathrm{U})$. Thus, $f(\mathtt{x}) \in Skyline(\mathcal{O}(\boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U})), \mathrm{U})$.
($\Leftarrow$) Assume that $\exists f(\mathtt{x}) \in Skyline(\mathcal{O}(\boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U})), \mathrm{U})$. So, $\forall \mathtt{y}_1 \in \boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U})$, $\mathtt{y}_1 \not\succ_{\mathrm{U}} \mathtt{x}$. From Def. 8: $\forall \mathtt{y}_2 \in \mathcal{L}_{\mathcal{I}} \setminus \boldsymbol{\ell}\text{-}Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U}), \exists \mathtt{y}_3 \in Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U}), \mathtt{y}_3 \succ_{\mathrm{U}} \mathtt{y}_2$. Thus, $\mathtt{y}_2 \not\succ_{\mathrm{U}} \mathtt{x}$. Therefore, $\mathtt{x} \in Sky(\mathcal{L}_{\mathcal{I}}, \mathrm{U})$.

The second step is performed using a skyline algorithm based on the `BNL` approach [2].

4 Related Work

Skylines vs Skypatterns. The notion of skyline queries [2, 3] has been recently integrated into pattern discovery to mine skypatterns [11]. Even if these notions seem similar, they correspond to very different extraction tasks. Skyline queries focus on the extraction of dominant tuples of a (point) database (T). The points (objects) are known in advance and then dominance test are applied. The skypattern mining task requires to mine patterns from a dataset (T) that must be Pareto-dominant for a given set of measures. Therefore, the latter problem is much harder since the search space for skypatterns is much larger than the search space for skylines: $O(\mathcal{L}_{\mathcal{I}}{=}2^{|\mathcal{I}|})$ instead of $O(|T|)$. Two methods have been designed for mining skypatterns: AETHERIS [11] is a two-step method that benefits from theoretical relationships between condensed representations

and skypatterns, while CP+SKY [15] mines skypatterns using dynamic CSPs. Finally, [12] provides a point-to-point comparison between these two approaches.

Skyline Cube vs Skypattern Cube. To offer the best possible response time for a subspace skyline query, skyline cubes (a.k.a SkyCube) were introduced independently by [17, 9]. They proposed several strategies to share skyline computation in different subspaces. Pei et al. proposed in [8] *Stellar*, which computes seed skylines groups in the full space, then extend it to build the final set of skyline groups and thus avoid the computation of skylines in all the subspaces. Similarly to the notions of skyline/skypattern, the skyline cube differs from the skypattern cube. A SkyCube tackles a point database looking for skyline point sets for a given set of dimensions. The skypattern cube computation has to deal with all the skypattern sets for a given set of measures. As seen in the previous paragraph, even if these notions are close, computing the skypattern cubes is much harder due to the huge search space. Two methods have been proposed to compute the skypattern cube. The first method, called as CP+SKY+CUBE [13], is based on a bottom-up approach and derivation rules exploiting the relation between the nodes in the lattice. The second method [14] proposes an approximation of the skypattern cube and then applies skyline cube mining in $|\mathcal{M}|$ dimensions on that approximation.

The Compressed Skyline Cube. Probably the closest previous work to our proposal is the so called *compressed skycube* (CSC) [16]. Its compression technique consists in storing for every subspace its partial skyline. It also supports concurrent subspace skyline queries in frequent updated databases. Our CSKYC can be seen as a reshaping of the CSC. However, the compressed skypattern cube computation is much harder due to the huge search space. Indeed, as shown previously, we need to extract patterns from a transactional dataset ($\mathcal{T}$) in order to determinate *proper* skypatterns for a given subspace. Other skycube summarization techniques have also been introduced. For instance, [1] proposed Hashcube, a structure based on bit-strings for storing the whole skycube. The work described in [6] proposed the negative skycube that returns subspaces where objects are not skylines.

5 Experimental Evaluation

This section evaluates constructing and querying the CSKYC on a real-life dataset and benchmarks. We compare the performances of the CSKYC with those of the original SKYC in terms of running-time and space storage, followed by query performance using CSKYC. The implementation of the different algorithms were carried out in C++. All experiments were conducted on a PC running Linux with a core i3 processor at 2.13 GHz.

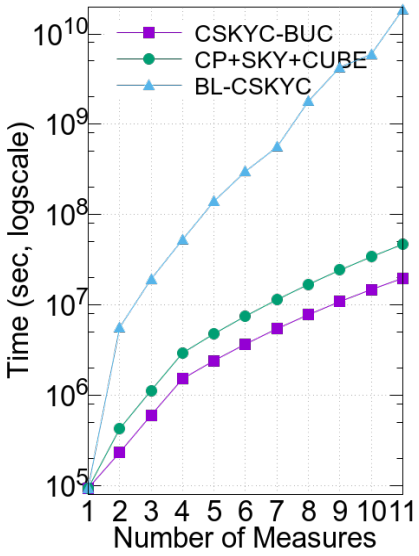
5.1 Compressed Skypattern Cubes for Mutagenicity Dataset

We performed experiments on a real-life dataset of large size extracted from mutagenicity data [5] (a major problem in risk assessment of chemicals). This dataset has $|\mathcal{T}|$=$6,512$ transactions encoding chemicals and $|\mathcal{I}|$=$1,073$ items encoding frequent closed subgraphs previously extracted from $\mathcal{T}$ with a 2% relative frequency threshold. Chemists use up to $|\mathcal{M}|$=11 measures, five of them are typically used in contrast mining

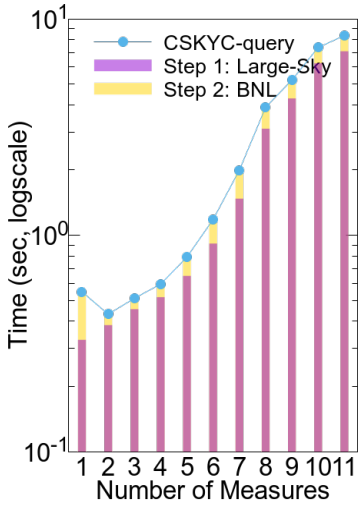
$\lvert\mathcal{M}\rvert$	$\sum_{\mathrm{U}\subseteq\mathcal{M}} \lvert Sky(\mathcal{L}_\mathcal{I},\mathrm{U})\rvert$ (2)	$\sum_{\mathrm{U}\subseteq\mathcal{M}} \lvert \boldsymbol{P}\text{-}Sky(\mathcal{L}_\mathcal{I},\mathrm{U})\rvert$ (3)	$1-\frac{(3)}{(2)}$	$2^{\lvert\mathcal{M}\rvert}-1$ (5)	(6)	$1-\frac{(6)}{(5)}$
1	338	338	0.00	1	1	0.00
2	753	493	0.35	3	1	0.67
3	1,280	784	0.39	7	4	0.43
4	1,983	1,123	0.43	15	6	0.60
5	2,982	1,525	0.49	31	11	0.65
6	4,526	1,990	0.56	63	20	0.68
7	7,146	2,484	0.65	127	32	0.75
8	12,015	2,924	0.76	255	51	0.80
9	21,773	3,246	0.85	511	82	0.84
10	42,386	3,462	0.92	1,023	149	0.85
11	87,374	3,853	0.96	2,047	281	0.86

(6) $\lvert\{\ \mathrm{U}\subseteq\mathcal{M}\ \mid\ \boldsymbol{P}\text{-}Sky(\mathcal{L}_\mathcal{I},\mathrm{U})\neq\emptyset\ \}\rvert$

(a) Space analysis.



(b) CPU-times.



(c) Query times.

Fig. 3: Results on Mutagenicity Dataset with $|\mathcal{M}| = 11$.

(e.g. growth rate) and allow to express different kinds of background knowledge. The other six measures are related to topological and chemical properties of the chemicals.

Space analysis. Fig. 3a shows the storage comparison of CSKYC to skypattern cubes of different dimensionality. Column 1 corresponds to the number of measures. Columns 2 and 3 report the total number of skypatterns for SKYC and CSKYC respectively. Column 4 gives their ratio. Columns 5 and 6 report the total number of subspaces for SKYC

and CSKYC respectively. Column 7 gives their ratio. For each $|\mathcal{M}| = k$, reported values in columns (2), (3), (5) and (6) represent the averages over all $\binom{11}{k}$ possible skypattern cubes. Overall, CSKYC achieves the best compression of the skypattern sets. The effect of duplicate elimination is greatly amplified for $|\mathcal{M}| \geq 6$. CSKYC achieves up to $20.6\times$ compression (in number of skypatterns) and permits using $4 - 7\times$ fewer subspaces. For $|\mathcal{M}| = 11$, the total number of proper skypatterns is $3,853$, while for SKYC the total number of skypatterns is $87,374$. This lead to a substantial gain greater than 95%.

CPU-time analysis. We compare our approach (`CSKYC-BUC`) with two methods: (i) a base-line method (`BL-CSKYC`) for computing CSKYC, and (ii) `CP+SKY+CUBE` proposed in [13] for computing SKYC. `BL-CSKYC` follows a bottom-up strategy: from the lower level, for each level and each subspace of the lattice, we compute its skypatterns, collect the skypatterns of its descendant subspaces, and then we remove all the duplicates. Fig. 3b shows the performance of the three methods according to the number of measures $|M|$. The scale is logarithmic. For `CSKYC-BUC` (resp. `CP+SKY+CUBE`)) and for $|M|$=k, the reported CPU-time is the average of CPU-times over all $\binom{11}{k}$ possible CSKYC (resp. SKYC). As we can see, `CSKYC-BUC` clearly outperforms `BL-CSKYC` by several orders of magnitude. This is particularly obvious for higher values of $|M|$ due to the reduced number of skypatterns involved in the construction. For $(2 \leq |\mathcal{M}| \leq 5)$, the average speed-up is 37.3. For $|\mathcal{M}| = 8$, there is an order of magnitude (speed-up value 213.15). For $|\mathcal{M}| = 11$, the speed-up value reaches 949. Finally, `CSKYC-BUC` is an average 2x faster than `CP+SKY+CUBE` for building the CSKYC.

Querying CSKYC. Evaluating the query performance of Algorithm 2, for $|M|$=k, is performed by dividing the total time to sequentially query every subspace by 2^k-1. Each query is extracted from the CSKYC. The reported CPU-time in Fig. 3c are the averages of CPU-times over all $\binom{11}{k}$ possible CSKYCs. By comparing the CPU-times for the two steps of Algorithm 2, overall, the `BNL` step is negligible as compared to the fist step. The scale is logarithmic. Experimental results show that query processing of the CSKYC is fast (less than 10 seconds for $|M|$=11).

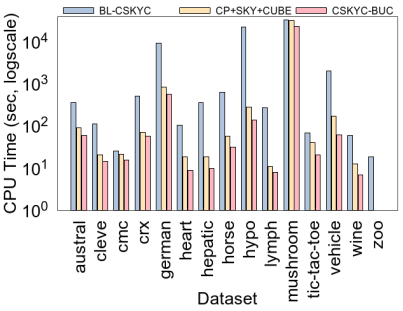
5.2 Compressed Skypattern Cubes for UCI Datasets

Experiments were carried out on 15 datasets from UCI benchmarks [4]. We considered 5 measures M=$\{$`freq`, `max`, `area`, `mean`, $\texttt{gr}_1\}$. In order to use measures using numeric values, like `mean`, we generated random values associated to attributes, each value being within the range $[0..1]$. Fig. 4 summarizes the results we obtained.

CPU-time analysis. Fig. 4a compares the performance of the three methods (with a graphical view). Cols. 1-4 give the characteristics of each dataset (name, number of items ($\mathcal{I}$), number of transactions ($\mathcal{T}$) and density). `CSKYC-BUC` clearly dominates the base-line method. On half of the datasets, there is an order of magnitude (speed-up value at least 11.78) (Col. 8). `CSKYC-BUC` is an average 2 times faster than `CP+SKY+CUBE`.

Space analysis. Fig. 4b compares, for each dataset, the number of proper skypatterns vs. the total number of skypatterns at each level of the cube. For each level i $(2 \leq i \leq 5)$, the corresponding summarization ratio is also depicted. Fig. 4c shows the graphical

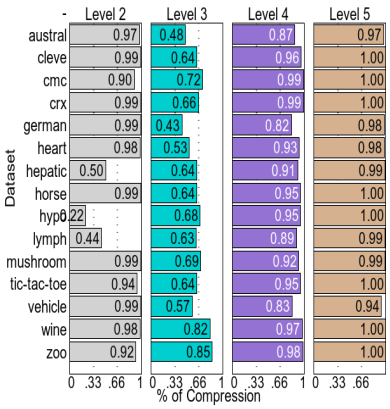
Dataset				CPU-Time			Speed-Up	
Name	$\mathcal{I}$	$\mathcal{T}$	Density	BL-CSKYC (5)	CP+SKY+CUBE (6)	CSKYC-BUC (7)	$\frac{(5)}{(7)}$	$\frac{(6)}{(7)}$
austral	55	690	0.272	6m04s	1m31s	**1m01s**	5.97	1.49
cleve	43	303	0.325	1m53s	21s	**15s**	7.53	1.40
cmc	28	1,474	0.357	26s	22s	**16s**	1.63	1.38
crx	59	690	0.269	8m40s	1m13s	**59s**	8.81	1.24
german	76	1,000	0.276	2h34m18s	14m03s	**9m36s**	16.07	1.46
heart	38	270	0.368	1m46s	19s	**9s**	11.78	2.11
hepatic	45	155	0.421	6m12s	19s	**10s**	37.20	1.90
horse	75	300	0.235	10m34s	58s	**32s**	19.81	1.81
hypo	47	3,163	0.389	6h13m57s	4m41s	**1m21s**	159.13	3.47
lymph	59	142	0.322	4m32s	11s	**8s**	34.00	1.38
mushroom	119	8,124	0.193	9h23m28s	8h54m43s	**6h32m15s**	1.44	1.36
tic-tac-toe	29	259	0.344	1m10s	41s	**21s**	3.33	1.95
vehicle	58	846	0.327	34m01s	2m55s	**1m03s**	32.40	2.78
wine	45	179	0.311	1m00s	13s	**7s**	8.57	1.86
zoo	43	102	0.394	19s	1s	**1s**	19.00	1.00



(a) Time Analysis.

$\lvert\mathcal{M}\rvert$	1	2			3			4			5		
Dataset	$Sky(\mathcal{L}_\mathcal{I},\mathrm{U})$ $\sum_{\lvert\mathrm{U}\rvert=1}$	$Sky(\mathcal{L}_\mathcal{I},\mathrm{U})$ $\sum_{\lvert\mathrm{U}\rvert=2}$ (3)	$\widetilde{Sky}(\mathcal{L}_\mathcal{I},\mathrm{U})$ $\sum_{\lvert\mathrm{U}\rvert=2}$ (4)	$1-\frac{(4)}{(3)}$	$Sky(\mathcal{L}_\mathcal{I},\mathrm{U})$ $\sum_{\lvert\mathrm{U}\rvert=3}$ (6)	$\widetilde{Sky}(\mathcal{L}_\mathcal{I},\mathrm{U})$ $\sum_{\lvert\mathrm{U}\rvert=3}$ (7)	$1-\frac{(7)}{(6)}$	$Sky(\mathcal{L}_\mathcal{I},\mathrm{U})$ $\sum_{\lvert\mathrm{U}\rvert=4}$ (9)	$\widetilde{Sky}(\mathcal{L}_\mathcal{I},\mathrm{U})$ $\sum_{\lvert\mathrm{U}\rvert=4}$ (10)	$1-\frac{(10)}{(9)}$	$Sky(\mathcal{L}_\mathcal{I},\mathrm{U})$ $\sum_{\lvert\mathrm{U}\rvert=5}$ (12)	$\widetilde{Sky}(\mathcal{L}_\mathcal{I},\mathrm{U})$ $\sum_{\lvert\mathrm{U}\rvert=5}$ (13)	$1-\frac{(13)}{(12)}$
austral	101,630	1,314	45	0.97	403	211	0.48	648	83	0.87	317	9	0.97
cleve	43,504	6,601	34	0.99	238	85	0.64	300	12	0.96	111	0	1.00
cmc	12,208	559	57	0.90	269	76	0.72	299	1	0.99	104	0	1.00
crx	107,742	5,216	58	0.99	307	105	0.66	437	1	0.99	215	0	1.00
german	1,173,279	35,083	60	0.99	620	352	0.43	1,198	210	0.82	610	13	0.98
heart	34,400	1,218	29	0.98	283	133	0.53	468	35	0.93	211	5	0.98
hepatic	9,598	189	95	0.50	537	194	0.64	762	71	0.91	343	4	0.99
horse	129,716	10,352	22	0.99	177	64	0.64	220	10	0.95	84	0	1.00
hypo	671,961	789	613	0.22	2,134	686	0.68	2,812	129	0.95	1,172	0	1.00
lymph	45,644	98	55	0.44	428	157	0.63	524	59	0.89	268	4	0.99
mushroom	650,965	96,164	45	0.99	216	66	0.69	307	24	0.92	131	1	0.99
tic-tac-toe	16,157	1,301	77	0.94	259	93	0.64	310	16	0.95	109	0	1.00
vehicle	141,974	15,901	69	0.99	480	208	0.57	827	144	0.83	421	24	0.94
wine	15,112	1,346	25	0.98	152	27	0.82	170	5	0.97	57	0	1.00
zoo	4,871	599	50	0.92	197	29	0.85	211	5	0.98	74	0	1.00

(b) Space Analysis.



(c) Summarization Ratios.

Fig. 4: Results on UCI datasets with $|\mathcal{M}| = 5$.

view of these ratios. For level 2, on most of the datasets, CSKYC achieves a very high summarization ratios (up to 99%). For level 3, these ratios mostly decrease since both levels (2 and 3) share most subspaces ($\binom{5}{2} + \binom{5}{3} = 20$ in total against 31 for the cube). Finally, for level 4 (resp. 5), CSKYC uses less storage than SKYC by at least 82% (resp. 94%) in size for all datasets we considered. Within these levels, there are almost no *proper* skypatterns since they have few subspaces ($\binom{5}{4} = 5$ for level 4 and $\binom{5}{5} = 1$) for level 5 and each one of these subspaces has a lot of descendant subspaces (14 for level 4 and 30 for level 5).

6 Conclusion

We have presented the compressed skypattern cube which concisely represents the skypattern cube and preserves its essential information. Compared to the original skypattern cube, the compressed skypattern cube has much less duplicates among susbpaces. We have provided an efficient algorithm to compute it and to query the skypattern set for any subspace. Our experimental study shows that CSKYC is particularly efficient

in terms of build time and space usage compared to the original skypattern cube. Another interesting property is its ability to efficiently provide the skypattern set of any subspace. As future work, we plan to investigate the incremental maintenance of the CSKYC by allowing to add and/or remove any measure.

References

1. Bøgh, K.S., Chester, S., Sidlauskas, D., Assent, I.: Hashcube: A data structure for space- and query-efficient skycube compression. In: CIKM. pp. 1767–1770 (2014)
2. Börzsönyi, S., Kossmann, D., Stocker, K.: The skyline operator. In: ICDE. pp. 421–430 (2001)
3. Chomicki, J., Godfrey, P., Gryz, J., Liang, D.: Skyline with presorting. In: ICDE. pp. 717–719 (2003)
4. Dua, D., Graff, C.: UCI machine learning repository (2017), http://archive.ics.uci.edu/ml
5. Hansen, K., Mika, S., Schroeter, T., Sutter, A., ter Laak, A., Steger-Hartmann, T., Heinrich, N., Müller, K.: Benchmark data set for in silico prediction of ames mutagenicity. JCIM 49(9), 2077–2081 (2009)
6. Hanusse, N., Wanko, P.K., Maabout, S.: Computing and summarizing the negative skycube. In: CIKM. pp. 1733–1742 (2016)
7. Mannila, H., Toivonen, H.: Levelwise search and borders of theories in knowledge discovery. Data Min. Knowl. Discov. 1(3), 241–258 (1997)
8. Pei, J., Fu, A.W., Lin, X., Wang, H.: Computing compressed multidimensional skyline cubes efficiently. In: ICDE. pp. 96–105 (2007)
9. Pei, J., Jin, W., Ester, M., Tao, Y.: Catching the best views of skyline: A semantic approach based on decisive subspaces. In: VLDB. pp. 253–264 (2005)
10. Pham, H., Lavenier, D., Termier, A.: Identifying genetic variant combinations using skypatterns. In: DEXA Workshops. pp. 44–48. IEEE Computer Society (2016)
11. Soulet, A., Raïssi, C., Plantevit, M., Crémilleux, B.: Mining dominant patterns in the sky. In: ICDM. pp. 655–664 (2011)
12. Ugarte, W., Boizumault, P., Crémilleux, B., Lepailleur, A., Loudni, S., Plantevit, M., Raïssi, C., Soulet, A.: Skypattern mining: From pattern condensed representations to dynamic constraint satisfaction problems. Artif. Intell. 244, 48–69 (2017)
13. Ugarte, W., Boizumault, P., Loudni, S., Crémilleux, B.: Computing skypattern cubes. In: ECAI. pp. 903–908 (2014)
14. Ugarte, W., Boizumault, P., Loudni, S., Crémilleux, B.: Computing skypattern cubes using relaxation. In: ICTAI. pp. 859–866 (2014)
15. Ugarte, W., Boizumault, P., Loudni, S., Crémilleux, B., Lepailleur, A.: Mining (soft-) skypatterns using dynamic CSP. In: CPAIOR. pp. 71–87 (2014)
16. Xia, T., Zhang, D.: Refreshing the sky: the compressed skycube with efficient support for frequent updates. In: SIGMOD Conference. pp. 491–502 (2006)
17. Yuan, Y., Lin, X., Liu, Q., Wang, W., Yu, J.X., Zhang, Q.: Efficient computation of the skyline cube. In: VLDB. pp. 241–252 (2005)