Dominance Analyses Reduction in Skyline Query Processing over Data Stream with Data Mining Technique

Mudathir Ahmed Mohamud
Department of Computer Science
Universiti Putra Malaysia
Serdang, Selangor, Malaysia
mudathir1100@gmail.com

Hamidah Ibrahim*
Department of Computer Science
Universiti Putra Malaysia
Serdang, Selangor, Malaysia
hamidah.ibrahim@upm.edu.my

Fatimah Sidi
Department of Computer Science
Universiti Putra Malaysia
Serdang, Selangor, Malaysia
fatimah@upm.edu.my

Siti Nurulain Mohd Rum
Department of Computer Science
Universiti Putra Malaysia
Serdang, Selangor, Malaysia
snurulain@upm.edu.my

Abstract

The database community has observed in the past two decades, the growth of research interest in skyline queries, which aim to report to users interesting objects—commonly known as skylines—based on their preferences. The identification of skyline objects becomes more challenging when skylines are to be identified from a collection of continuously generated input data streams. In this paper, we proposed the *Dominance Analyses Reduction* (*DAR*) framework, which aims at addressing the issues of redundant dominance analyses that arise while determining skylines over data stream. Dominance analyses are repeated for objects that are in the overlapped frames of two windows and for pairs of objects that later reappear in the stream. DAR employs the Apriori algorithm, one of the most prevalent data mining algorithms, to identify the frequently occurring dominance analyses. Instead of conducting the dominance analyses again, their results are stored and utilised in the subsequent derivation of skylines. The DAR framework has been validated through several experiments. Its results exhibit significant reduction in the number of pairwise comparisons at both object and dimension levels and execution time.

CCS Concepts

• **Theory of computation** → Theory and algorithms for application domains; Database theory; Data structures and algorithms for data management.

Keywords

Dominance analyses, Skyline query processing, Data stream, Data mining

*Corresponding Author.

ICICM 2024, Paris, France

ACM ISBN 979-8-4007-1747-5/2024/11
https://doi.org/10.1145/3711609.3711614

ACM Reference Format:
Mudathir Ahmed Mohamud, Hamidah Ibrahim, Fatimah Sidi, and Siti Nurulain Mohd Rum. 2024. Dominance Analyses Reduction in Skyline Query Processing over Data Stream with Data Mining Technique. In *2024 The 14th International Conference on Information Communication and Management (ICICM 2024), November 06–08, 2024, Paris, France.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3711609.3711614

1 Introduction

Recently, skyline queries which rely on Pareto dominance have attracted a lot of interest from the database community. Since the skyline operator was first introduced by [1], a plethora of skyline algorithms have been proposed [1–9]. The operator [1] selects objects in a dataset that are not dominated by other objects, thereby filtering the collection of objects in the dataset. If an object is as good as the other object in all dimensions and better in at least one dimension, then the object is said to dominate the other object. It is therefore necessary to perform exhaustive *pairwise comparisons* or *dominance analyses* (*tests*) among the objects in the dataset in order to determine the best and most desired objects, or *skylines*.

The paradigm shift from static data to streaming data has recently attracted the attention of the database community, primarily due to the inefficiencies of traditional database techniques in handling the unique characteristics of data streams. Skyline query processing is no exception where skyline objects must be updated constantly as new objects arrive and existing objects expire over time. Processing skyline queries over data streams is challenging mainly because objects in the stream arrive online while the data stream is potentially unbounded in size. In stream processing, sliding windows (or windows in short) are introduced to allow objects to be processed in small, manageable chunks over a predefined period of time. Therefore, processing skyline queries over data stream requires skylines to be reported at each window. However, redundant dominance analyses occur when objects are within the overlapping frame of two windows as they are subjected to the same dominance analyses in both windows. Also, it is unwise to repeat the dominance analyses on pairs of objects that reappear in the stream at a later time.

Table 1: Examples of skyline objects

Object	Price	Distance	$t_{arr}(o_i)$
o_1	200	0.5	1
o_2	150	2	2
o_3	25	15	3
o_4	125	3.5	4
o_5	100	5	6
o_6	75	8	7
o_7	115	2	8
o_8	176	4	10
o_3	25	15	12
o_9	60	15	14
o_{10}	186	2	16
o_5	100	5	18

In this paper, we proposed a framework called *Dominance Analyses Reduction* (*DAR*), which aims at addressing the issues of redundant dominance analyses that arise while determining skylines over data stream. *DAR* employs the Apriori algorithm, one of the prevalent algorithms in data mining, to identify the most frequently occurring dominance analyses. The results of these frequent dominance analyses are saved and used in later skyline computation. Following these, in this paper, we make the following contributions: (i) We discuss in detail the issues of redundant dominance analyses over data stream and emphasise the need to avoid them. (ii) We proposed the *Dominance Analyses Reduction* (*DAR*) framework, which employs the Apriori algorithm to identify the most frequently occurring dominance analyses; that are then utilised in deriving skylines over data stream. (iii) We conduct several experiments and evaluate the performance of *DAR* framework against the conventional skyline algorithm with regard to the number of pairwise comparisons and execution time.

The structure of this paper is as follows. The motivation behind this work is presented in Section 2 while the review on works related to the study is given in Section 3. Meanwhile, the related definitions and notations to clarify the proposed framework, *DAR*, are provided in Section 4. The *DAR* framework is explained in Section 5, while its performance study is reported in Section 6. The paper is concluded in Section 7.

2 Motivation

In this section, we explain the significance of avoiding unnecessary dominance analyses (tests) that arise while determining skylines over data stream. The notion of skyline queries is to find a set of objects that is not dominated by any other objects by comparing their corresponding dimensions. For an example, consider the objects presented in Table 1. Assume a user is interested in looking for apartments to rent that are as cheap as possible and as close as possible to the city centre. Applying the skyline algorithm on the given samples of objects would retrieve the following skyline results, $S = \{o_1,\ o_3,\ o_5,\ o_6,\ o_7\}$. $o_2(150,\ 2)$ for instance, is not a skyline as it is being dominated by $o_7(115,\ 2)$ since o_7 has a lower price value than o_2 with both having the same distance to the city center.

Processing skyline queries over data stream imposes a number of challenges that could result in high computation time due to unnecessary dominance analyses that are conducted. In a data stream, every object o_i has a timestamp indicating the arrival time, $t_{arr}(o_i)$, of the object in the stream. Consider the following skyline query; "a user may ask the most preferable apartment deals advertised and want to update the results every 10 seconds"; the skyline objects for the given query are derived based on the objects that fall within the timeframe $0 - 10$ (w_1) and $7 - 20$ (w_2); where w_1 and w_2 are the windows of the data stream as shown in Figure 1. This means the skyline algorithm is executed at the end of each window, i.e. at time $t = 10$ and $t = 20$, respectively; that is when all objects of a given window are identified. With the assumption that the set of objects, $O_{w_k} = \{o_1, o_2, \ldots, o_n\}$, with each object having m dimensions falls within the window w_k, the average number of pairwise comparisons performed between objects (*npco*) is given by the Equation (1); while the average number of pairwise comparisons performed based on dimensions (*npcd*) is given by the Equation (2) [10]. If every object has 16 dimensions, then the *npcd* of w_1 and w_2, shown in Figure 1, are $npcd_{w_1} = 16\frac{8(8-1)}{2} = 448$ and $npcd_{w_2} = 16\frac{7(7-1)}{2} = 336$; while the *npco* of w_1 and w_2 are 28 and 21, respectively.

$$npco_{w_k} = \frac{n\,(n-1)}{2} \tag{1}$$

$$npcd_{w_k} = m\frac{n\,(n-1)}{2} \tag{2}$$

There are two cases in which unnecessary dominance analyses occur over data stream as follows: (i) Objects that fall inside the overlapping frame of two windows. For instance, the objects that appear at 7, 8, and 10 seconds of Figure 1 fall in both windows, w_1 and w_2, and these objects are analysed in the computation of skylines of both windows. This means that 48×2 pairwise comparisons based on dimensions are being conducted and half of these comparisons are unnecessary. (ii) Pair of objects that appear again in the stream at a later time. The objects o_3 and o_5, for example, arrived in window w_1 at 3 and 6 seconds, respectively; they also appeared in window w_2 at 12 and 18 seconds, respectively. If these objects have already been compared in w_1, then comparing them again in w_2 will incur unnecessary dominance analysis. A naive approach to this problem is storing every dominance analysis result that can be referred to in a later dominance analyses. However, some objects will only appear once in the stream, making this approach unrealistic.

3 Related work

Since the introduction of the skyline operator by [1], many variations of skyline algorithms have been developed. Among the noteworthy skyline algorithms are *Divide-and-Conquer* (*D&C*), *Block Nested Loop* (*BNL*) [1], *Bitmap and Index* [2], *Sort Filter Skyline* (*SFS*) [3], *Nearest Neighbor* (*NN*) [4], *Linear Elimination Sort Skyline* (*LESS*) [5], *Branch and Bound Skyline* (*BBS*) [6], *Lattice Skyline* (*LS*) [7], *Implicit Preference Order Tree* [8], and *Sort and Limit Skyline Algorithm* (*SaLSa*) [9]. In recent past, focus has been given to resolving issues related to the skyline computation over data streams, in which several different approaches have been established [11 – 22].

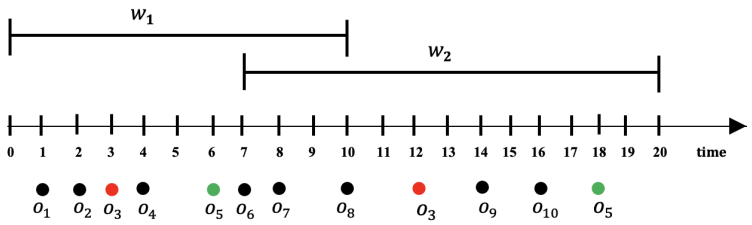

Figure 1: Examples of objects in a data stream

The work in [11], for instance, proposed algorithms that continuously monitor the incoming objects and incrementally maintain the skylines; while [12] introduced the continuous time-interval skyline operator which continuously computes the current skyline and an algorithm called *LookOut* which efficiently evaluates the skyline queries over a data stream. Meanwhile, [21] proposed a framework that periodically updated an index structure which is then employed in processing the skyline queries.

On the other hand, several extensions to skyline query have been made for data stream primarily to accommodate diverse user needs and applications. For instance, the work in [13] focuses on processing skyline queries on any subset of k dimensions based on the full space skyline, while [16] focuses on *reverse skyline queries* that have been used in numerous real-world applications such as business planning, market analysis, and environmental monitoring. Meanwhile, [20] emphasises on *ρ-dominant skyline query* which is suitable for the application on fast deciding data stream and extends it into n-of-N ρ-dominant and ($n1$,$n2$)-of-$N\rho$-dominant skyline query. Furthermore, a *representative skyline* containing k skyline points that can represent its corresponding full skyline is studied in [18]. Another work by [17], focuses on skyline group problem that finds k-item groups that cannot be dominated by any other k-item group.

Additionally, some works investigate alternative approaches to data stream implementation. For example, [14] focuses on distributed data streams, which originate from multiple horizontally split data sources; [19] argued that most of the past works proposed sequential algorithms for continuous skyline queries and thus utilized the parallel implementation for multicore architectures in data stream, and [22] manages the spatial-keyword skyline queries over geo-textual streams to continuously obtain skyline results. Additionally, [15] has investigated effective methods for handling multiple skyline queries and presented the *FAST* filtering technique. Despite the large number of works that have been published on skyline queries over data stream, the issues related to unnecessary dominance analyses over data stream have not been fully investigated; which this paper attempts to explore.

4 Prelimineries

In this section, we present the necessary definitions and introduce the notations that are used throughout this paper. We assume a database, D, with n objects, $D = \{o_1, o_2, \ldots, o_n\}$, with each object having m dimensions, $d = \{d_1, d_2, \ldots, d_m\}$. Throughout this paper, we assume that smaller values are preferable than larger values.

Definition 1 *Dominance*: The object $o_i \in D$ is said to dominate the object $o_j \in D$ denoted by $o_i \prec o_j$ where $i \neq j$ if and only if the following condition holds: $\forall d_k \in d,\ o_i.d_k \leq o_j.d_k \wedge \exists d_l \in d,\ o_i.d_l < o_j.d_l$. The process of comparing the objects o_i and o_j in pair to decide which of these objects is preferred is generally called *pairwise comparison* or *dominance analysis* (*test*). In this paper, we use the notation $\alpha(o_i, o_j)$ to represent the dominance analysis conducted between the objects o_i and o_j.

Definition 2 *Dominance Relationship*: Performing $\alpha(o_i, o_j)$ will yield one of the following results: (i) $o_i \prec o_j$ – o_i dominates o_j, (ii) $o_j \prec o_i$ – o_j dominates o_i, or (iii) $o_i \not\prec o_j$ and $o_j \not\prec o_i$ – both o_i and o_j do not dominate each other. The result of the dominance analysis presents the *dominance relationship* between the objects being compared.

Definition 3 *Symmetrical Property of Dominance Analysis*: The dominance analysis is symmetrical which means $\alpha(o_i, o_j)$ and $\alpha(o_j, o_i)$ are the same and will yield the same dominance relationship result. Hence, the order of the objects in the dominance analyses is insignificant.

Definition 4 *Skyline*: An object $o_i \in D$ is a skyline of D if there is no other objects, say $o_j \in D$ where $i \neq j$, that dominates o_i. We use the notation S_D to denote the set of skyline objects of database D.

Definition 5 *Sliding Window*: A sliding window w_k is denoted by $w_k[lb, ub]$ where lb and ub represent the lower-bound and upper-bound values, respectively, in units of time, t. The lb and ub values are derived based on the *range* and *slide* values of a given skyline query.

Definition 6 *Window Skyline*: An object $o_i \in O_{w_k}$ is the window skyline of w_k where $O_{w_k} = \{o_1, o_2, \ldots, o_n\}$, i.e. the set of objects of w_k, if there is no other object, say $o_j \in O_{w_k}$ where $i \neq j$, that dominates o_i. The notation S_{w_k} is used to denote the set of skyline objects of window w_k.

Definition 7 Redundant Dominance Analyses in Different Windows: Let $\alpha(o_i, o_j)^{t_k}_{w_x}$ represents the dominance analysis performed between o_i and o_j at time t_k in w_x. If $\alpha(o_i, o_j)^{t_l}_{w_y}$ (or $\alpha(o_j, o_i)^{t_l}_{w_y}$, see Definition 3) is performed where $t_l > t_k$ and $w_x \neq w_y$, then the dominance analyses of $\alpha(o_i, o_j)^{t_k}_{w_x}$ and $\alpha(o_i, o_j)^{t_l}_{w_y}$ are said to be redundant; in which the α at t_l in w_y is unnecessary. Regardless the time, the results of the dominance analyses are the same. For instance, $\alpha(o_3, o_5)^{t_{10}}_{w_1}$ and $\alpha(o_3, o_5)^{t_{20}}_{w_2}$ both result in $o_3 \not\prec o_5$ and $o_5 \not\prec o_3$. This applies to (i) the pair of objects with more than one arrival time, $t_{arr}(o_i)$ (like o_3 and o_5) and each $t_{arr}(o_i)$ falls in different windows of the stream and (ii) the pair of objects that fall inside the overlapping frame of two windows (like o_6, o_7 and o_8).

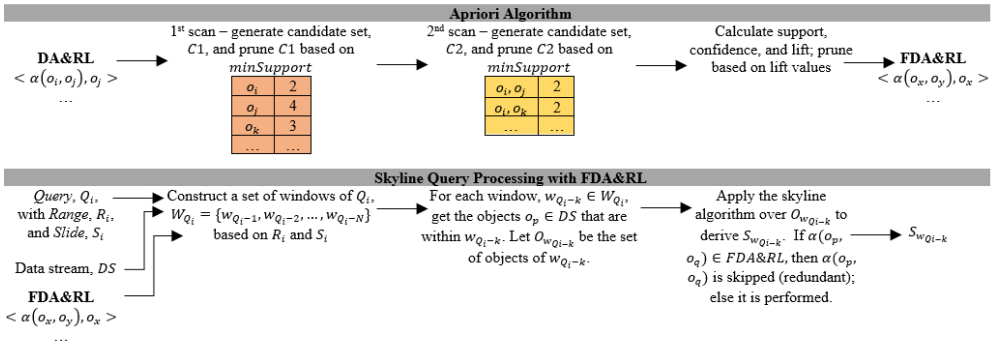

Figure 2: The DAR framework

Definition 8 Redundant Dominance Analyses in the Same Window: Let $\alpha(o_i, o_j)^{t_k}_{w_x}$ represents the dominance analysis performed between o_i and o_j at time t_k in w_x; with each object having $t_{arr}(o_i)$ and $t_{arr}(o_j)$, respectively. If $\alpha(o_i, o_j)^{t_k}_{w_x}$ is to be performed again with $t_{arr}(o_i')$ and $t_{arr}(o_j')$, respectively, where $t_{arr}(o_i') > t_{arr}(o_i)$ and $t_{arr}(o_j') > t_{arr}(o_j)$, then these dominance analyses are said to be redundant. Regardless the different arrival times of these objects, the results of the dominance analyses are the same. This applies to the pair of objects with more than one arrival time, $t_{arr}(o_i)$, that fall in the same window of the stream; which normally occur when the size of the sliding window is large.

5 The proposed skyline query processing with frequent dominance analysis and relationship list (FDA&RL)

This section presents our proposed framework named *Dominance Analyses Reduction (DAR)* that are designed mainly to deal with the issues of redundant dominance analyses that occur over data stream. *DAR* consists of two main phases namely: *Apriori Algorithm* phase and *Skyline Query Processing with FDA&RL* phase as shown in Figure 2. These phases are further elaborated in the following paragraphs.

The main aim of the *Apriori Algorithm* phase is to identify the dominance analyses that are frequently performed over the stream which is achieved by analysing a list of dominance analyses (DA&RL) that have been performed in earlier skyline query processing. The DA&RL list contains the pairs of objects that have been compared together with their dominance relationships, and has the following structure $\langle \alpha(o_i, o_j),\ r \rangle$ where r is the dominance relationship as defined in *Definition* 2. This phase follows the standard steps of Apriori Algorithm [23] which are briefly explained below:

(i) Create a table called candidate set, $C1$, containing the objects present in the DA&RL with their support count, *supCount*; compare the *supCount* of each object to the minimum support, *minSupport*; and remove those objects in $C1$ whose $supCount < minSupport$. Here, *minSupport* is set to 2.

(ii) Create a table called candidate set, $C2$, containing *supCount*, of each pair of objects in $C1$; compare the *supCount* of each pair of objects to the *minSupport*; and remove those pairs of objects in $C2$ whose $supCount < minSupport$. Since each dominance analysis only involves two objects, hence no further iteration is performed. At this stage, the frequent dominance analyses are discovered.

(iii) The following equations are then employed to generate the strong association rules (dominance analyses) that are then saved in the FDA&RL:

$$Support = \frac{supCount\ (o_i, o_j)}{|DA\&RL|} \tag{3}$$

$$Confidence = \frac{supCount\ (o_i, o_j)}{supCount\ (o_i)} \tag{4}$$

$$Lift = \frac{Support}{Support\ (o_i) \times Support\ (o_j)} \tag{5}$$

In general *Support* as presented by Equation (3) indicates how frequently is the pair of objects, o_i and o_j, appears in the DA&RL and *Confidence* as presented in Equation (4) indicates the number of times the $o_i \rightarrow o_j$ is found to be true. On the other hand, $Lift$ as shown in Equation (5) is used to compare the observed confidence with expected confidence, i.e. how many times $o_i \rightarrow o_j$ is expected to be true. A value of $lift$ greater than 1 means high associations between the objects o_i and o_j. With this regard, those dominance analyses with pair of objects having $lift$ greater than 1 are saved in the FDA&RL to be utilised in the second phase of the *DAR* framework.

The *Skyline Query Processing with FDA&RL* phase computes the skylines of a given skyline query, Q_i, over a data stream, DS. In the process of deriving the skylines, the FDA&RL produced in the previous phase is utilised. The steps performed in this phase are as follows:

(i) Construct a set of windows based on the range, R_i, and slide, S_i, of Q_i. Let the set of windows be $W_{Q_i} = \{w_{Q_i-1}, w_{Q_i-2}, \ldots, w_{Q_i-N}\}$, with each window w_{Q_i-k} denoted by $w_{Q_i-k}[lb, ub]$ where lb and ub represent the lower-bound and upper-bound values, respectively, in units of time as defined in *Definition* 5.

(ii) For each window, $w_{Q_i-k} \in W_{Q_i}$, objects of the stream, $o_p \in DS$, whose arrival time, $t_{arr}(o_p)$, is within the $[lb, ub]$

of the window are identified. Let $O_{w_{Q_i-k}}$ be the set of objects of w_{Q_i-k}.

(iii) Apply the skyline algorithm over $O_{w_{Q_i-k}}$ to derive the skylines of w_{Q_i-k}, $S_{w_{Q_i-k}}$, as defined in *Definition* 6. This requires pairs of objects to be compared to decide which of these objects is preferred as explained in *Definition* 1. For each $\alpha(o_i, o_j)$ to be conducted, if $\alpha(o_i, o_j) \in$ FDA&RL, then the $\alpha(o_i, o_j)$ is skipped while its dominance relationship r is recorded. This is to avoid unnecessary dominance analyses as clearly presented in *Definition* 7 and *Definition* 8. However, if $\alpha(o_i, o_j) \notin$ FDA&RL, then the dominance analysis over these two objects is performed. Note that the symmetrical property of dominance analyses as presented in *Definition* 3 indicates that if it is found that $\alpha(o_i, o_j) \in$ FDA&RL, then $\alpha(o_j, o_i)$ is also considered as a member of FDA&RL.

(iv) The set of skylines of the window w_{Q_i-k}, $S_{w_{Q_i-k}}$, is then displayed at time, $t = ub$ of the window.

Algorithm 1 depicts the conventional skyline algorithm (*CSA*), while Algorithm 2 presents the skyline algorithm with the FDA&RL.

Algorithm 1 CSA

```
Input:  The set of objects of w_{Q_i-k}, O_{w_{Q_i-k}} = {o_1, o_2, ..., o_n}
Output: The set of skylines of the window w_{Q_i-k}, S_{w_{Q_i-k}}
BEGIN
  FOR each object, o_i, in O_{w_{Q_i-k}} DO
    FOR each object, o_j, in O_{w_{Q_i-k}} where i ≠ j DO
    BEGIN
      IF o_i ≺ o_j THEN delete o_j from O_{w_{Q_i-k}}
      ELSE
      IF o_j ≺ o_i THEN delete o_i from O_{w_{Q_i-k}}
    END
    S_{w_{Q_i-k}} = O_{w_{Q_i-k}}
END
```

Algorithm 2 SA with FDA&RL

```
Input:  The set of objects of w_{Q_i-k}, O_{w_{Q_i-k}} = {o_1, o_2, ..., o_n},
FDA&RL
Output: The set of skylines of the window w_{Q_i-k}, S_{w_{Q_i-k}}
BEGIN
  FOR each object, o_i, in O_{w_{Q_i-k}} DO
    FOR each object, o_j, in O_{w_{Q_i-k}} where i ≠ j DO
    BEGIN
      IF α(o_i, o_j) ∈ FDA&RL THEN
      BEGIN
        IF r = o_i THEN delete o_j from O_{w_{Q_i-k}}
        ELSE
        IF r = o_j THEN delete o_i from O_{w_{Q_i-k}}
      END
      ELSE
      BEGIN
        IF o_i ≺ o_j THEN delete o_j from O_{w_{Q_i-k}}
        ELSE
        IF o_j ≺ o_i THEN delete o_i from O_{w_{Q_i-k}}
      END
    END
  S_{w_{Q_i-k}} = O_{w_{Q_i-k}}
END
```

6 Result and discussion

To evaluate the efficiency of the proposed framework, *DAR*, in processing skyline queries over data stream, several experiments were designed. These experiments were conducted on Intel Core i5 PC with 1.80GHz processor and 8GB memory. The implementation of *DAR* was done in Python programming language running on a 64bit Windows 11. Each experiment was run 10 times and we report the average value of these runs. Each run lasted for 1000 seconds with three 400-second sliding windows that overlapped by 100 seconds. The performance measurement used in our experiments is number of pairwise comparisons based on object and dimension. Since this study is the first attempt that investigates the issues of redundant dominance analyses over data stream, thus we compare the *DAR* framework against the conventional skyline algorithm, herein called the *CSA*. We also compare the execution time of both *DAR* and *CSA*.

As *DAR* relies on the FDA&RL generated by the Apriori algorithm, we prepared the DA&RL by running the skyline algorithm over objects that are randomly distributed in three sliding windows, each lasting 100 seconds. Figure 3 presents the numbers of dominance analyses recorded in the DA&RL and FDA&RL. Figure 3(a) shows that when the number of objects, $|n|$, increases while the number of dimensions, $|d|$ is fixed to 10; in most cases there is a slight increment in the number of dominance analyses saved in the DA&RL. Nonetheless, the number of dominance analyses in the FDA&RL gets lower at the point when $|n|$ is 10K. This is because when $|n|$ increases, there are more distinct objects; hence the number of pairs of objects having $lift > 1$ is fewer. On the other hand, the numbers of dominance analyses saved in the DA&RL and FDA&RL for varying $|d|$ are the same; that are 1499959 and 39991, respectively, as shown in Figure 3(b). This is because $|n|$ is fixed to 100K; hence the same set of objects is used but at each run every object is associated with different number of dimensions, $|d|$.

We have carried out three primary analyses. In the first analysis, the effect of the number of objects, $|n|$, in the stream on the performance of *DAR* and *CSA* is investigated. The parameter settings of the synthetic dataset are as follows: the number of objects, $|n|$, is varied from 2K to 1M, the number of dimensions, $|d|$, is fixed to 10, while the percentage of duplicate objects, δ, is set to 50%. Figures 4(a1) and 4(a2) present the performance of *DAR* and *CSA* with regard to the number of pairwise comparisons based on object and dimension, respectively. These figures clearly demonstrate that both techniques observed an increase in the number of pairwise comparisons as $|n|$ increased. Although, in most cases, the number of pairwise comparisons of *DAR* is lower than *CSA* with an average of 97.73% of reduction; yet, the gap between these two approaches gets narrower as $|n|$ increases. This is because there are more distinct objects in the stream, which leads to fewer pairs of objects having $lift > 1$. As a result, there are fewer dominance analyses saved in the FDA&RL.

In the second analysis, the effect of the number of dimensions, $|d|$, in the stream on the performance of *DAR* and *CSA* is investigated. The parameter settings of the synthetic dataset are as follows: the

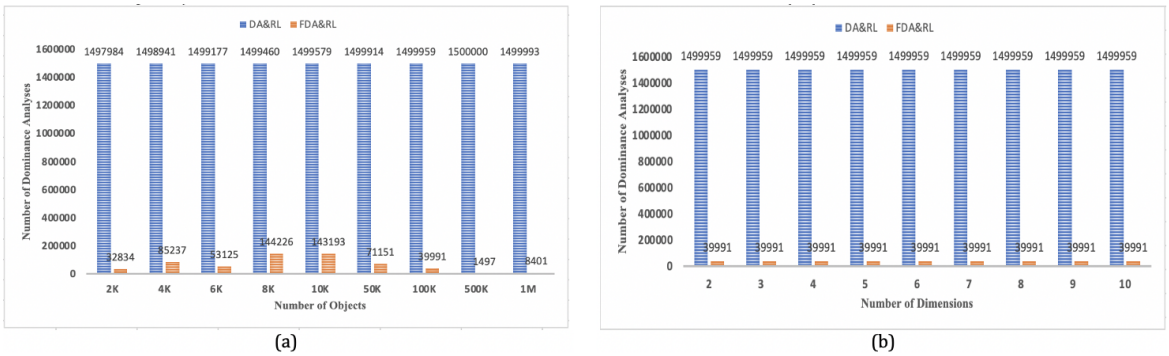

Figure 3: The DA&RL and FDA& RL with varying (a) number of objects, $|n|$ (b) number of dimensions, $|d|$

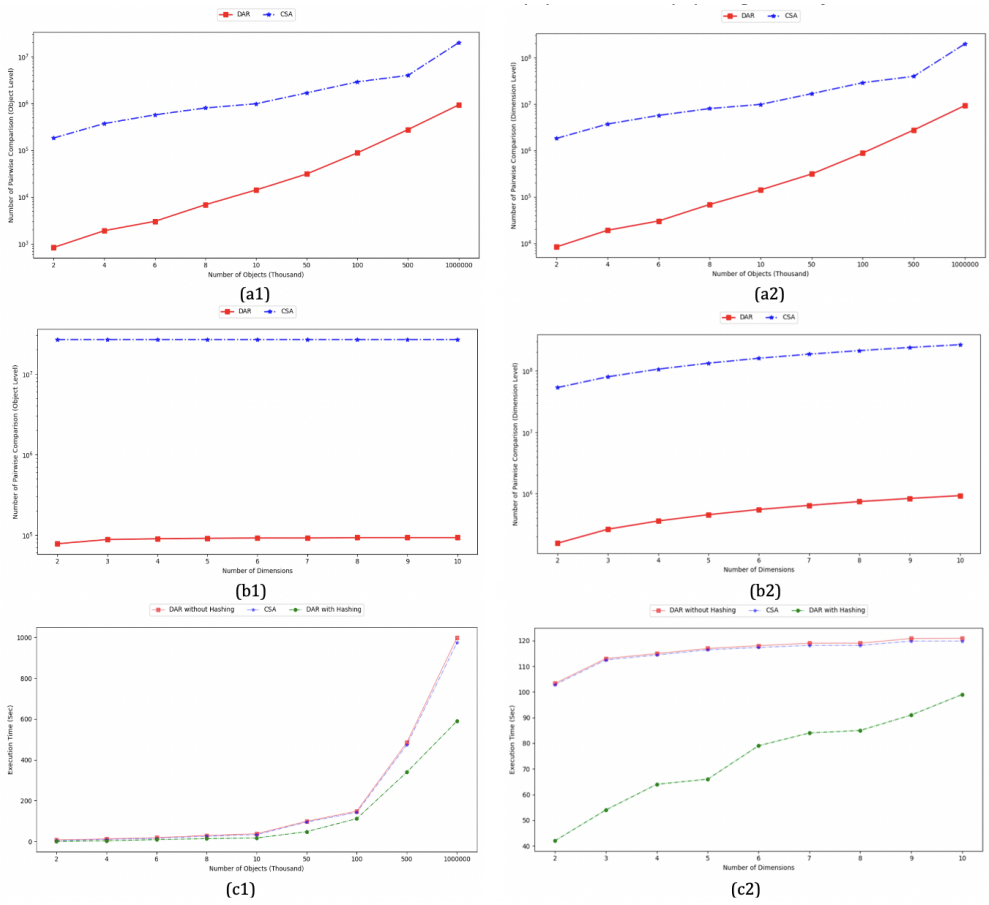

Figure 4: The number of pairwise comparisons with varying (a) $|n|$, (a1) based on object (a2) based on dimension; (b) $|d|$, (b1) based on object (b2) based on dimension; the execution time with varying (c1) $|n|$, (c2) $|d|$

number of dimensions, $|d|$, is varied from 2 to 10, the number of objects, $|n|$, is fixed to 100K, while the percentage of duplicate objects, δ is set to 50%. Figures 4(b1) and 4(b2) present the performance of *DAR* and *CSA* with regard to the number of pairwise comparisons based on object and dimension, respectively. These figures clearly demonstrate that both techniques observed a little increase in the number of pairwise comparisons based on objects and a slightly greater increment based on dimension as $|d|$ increased. This is because each run uses the same set of objects but with different $|d|$, resulting in nearly identical dominance analyses for each run. Nonetheless, in all cases, the number of pairwise comparisons of *DAR* is lower than *CSA* with an average of 99.66% of reduction.

In the third analysis, the performance of *DAR* and *CSA* is investigated with regard to the execution time. We use the same parameter settings as described above and the results are presented in figures 4(c1) and 4(c2). Since the *DAR* framework depends on the FDA&RL which typically contains thousands of dominance analyses, hence we have decided to employ the hashing technique to efficiently store and retrieve the relevant dominance analysis for quick access. Both the figures 4(c1) and 4(c2) show that the execution times of *DAR without hashing* and *CSA* are nearly equal. This is due to the fact that even though *DAR* reduces the number of pairwise comparisons, it still requires time to search and retrieve the dominance analyses from the FDA&RL. However, the execution time is reduced when hashing technique is applied as shown by *DAR with hashing* in both figures. On average, the amount of execution time is reduced to 48.95% and 36.61% for varied $|n|$ and varied $|d|$, respectively.

7 Conclusion

In this paper, we proposed the *Dominance Analyses Reduction* (*DAR*) framework, which aims at addressing the issues of redundant dominance analyses that arise while determining skylines over data stream. To identify the frequently occurring dominance analyses, *DAR* employs the Apriori algorithm. The dominance analyses are saved in the FDA&RL which is then utilised in the subsequent derivation of skylines. Several experiments have been conducted and the results show significant reduction in the number of pairwise comparisons at both object and dimension levels and execution time. Further enhancements that can be made based on the findings presented in the paper include: (i) investigating other indexing techniques besides hashing that could efficiently store and retrieve the relevant dominance analysis for quick retrieval and (ii) analysing how well other data mining algorithms like Eclat and FP-growth in handling the issues of redundant dominance analyses by generating the prominent association rules.

Acknowledgments

This work was supported by the Ministry of Higher Education Malaysia under the Fundamental Research Grant Scheme (FRGS/1/2020/ICT03/UPM/01/1) and the Universiti Putra Malaysia.

References

[1] Börzsöny S., Kossmann D., and Stocker K. 2001. The skyline operator. In Proceedings of the 17th. International Conference on Data Engineering. IEEE, Heidelberg, Germany. 421–430. https://doi.org/10.1109/ICDE.2001.914855

[2] Kian-Lee Tan, Pin-Kwang Eng, and Beng Chin Ooi. 2001. Efficient progressive skyline computation. In Proceedings of the 27th. International Conference on Very Large Data Bases. ACM, Roma, Italy. 301–310.

[3] Chomicki Jan, Parke Godfrey, Jarek Gryz, and Dongming Liang. 2003. Skyline with presorting. In Proceedings of the International Conference on Data Engineering (ICDE). IEEE, Bangalore, India. 717–719. https://doi.org/10.1109/ICDE.2003.1260846

[4] Donald Kossmann, Frank Ramsak, and Steffen Rost. 2002. Shooting stars in the sky: an online algorithm for skyline queries. In Proceedings of the 28th. International Conference on Very Large Data Bases. ACM, Hong Kong SAR, China. 275–286. https://doi.org/10.1016/B978-155860869-6/50032-9

[5] Parke Godfrey, Ryan Shipley, and Jarek Gryz. 2005. Maximal vector computation in large data sets. In Proceedings of the International Conference on Very Large Data Bases. ACM, Trondheim, Norway. 229–240.

[6] Dimitris Papadias, Yufei Tao, Greg Fu, and Bernhard Seeger. 2003. An optimal and progressive algorithm for skyline queries. In Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '03). ACM, San Diego, California, USA. 467–478. https://doi.org/10.1145/872757.872814

[7] Michael D. Morse, Jignesh M. Patel, and Jagadish H.V. 2007. Efficient skyline computation over low-cardinality domains. In Proceedings of the 33rd. International Conference on Very Large Data Bases. ACM, Vienna, Austria. 267–278.

[8] Chen, A. 2011. Dynamic implicit strict partially ordered sets. In Proceedings of the Fourth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT 2011). IEEE, Stevens Point, Wi, USA. 143–146. https://doi.org/10.1109/ICADIWT.2011.6041416

[9] Ilaria Bartolini, Paolo Ciaccia, and Marco Patella. 2006. Salsa: computing the skyline without scanning the whole sky. In Proceedings of the International Conference on Information and Knowledge Management (CIKM). ACM, Arlington, Virginia, USA. 405–414. https://doi.org/10.1145/1183614.1183674

[10] Dehaki, G.B., Ibrahim, H., Alwan, A.A., Sidi, F., and Udzir, N.I. 2021. Efficient skyline computation over an incomplete database with changing states and structures. *IEEE Access*, vol. 9, 2021, 88699–88723.

[11] Tao Y. and Papadias D. 2006. Maintaining sliding window skylines on data streams. IEEE Transactions on Knowledge Data Engineering, Vol. 18, No. 3, 377–391.

[12] Morse M., Patel J.M., and Grosky W.I. 2007. Efficient continuous skyline computation. Information Sciences, Vol. 177, No. 17, 3411–3437.

[13] Huang Z., Sun S., and Wang W. 2010. Efficient mining of skyline objects in subspaces over data streams. Knowledge and Information Systems, Vol. 22, No. 2, 159–183.

[14] Sun S., Huang Z., Zhong H., Dai D., Liu H., and Li J. 2010. Efficient monitoring of skyline queries over distributed data streams. Knowledge and Information Systems, Vol. 25, No. 3, 575–606.

[15] Lee Y.W., Lee K.Y., and Kim M.H. 2013. Efficient processing of multiple continuous skyline queries over a data stream. Information Sciences, Vol. 221, 316–337.

[16] Xin J., Wang Z., Bai M., and Wang G. 2015. Reverse skyline computation over sliding windows, Mathematical Problems in Engineering, Vol. 2015, 1–19.

[17] Guo X., Li H., Wulamu A., Xie Y., and Fu Y. 2016. Efficient processing of skyline group queries over a data stream. Tsinghua Science and Technology, Vol. 21, No. 1, 29–39.

[18] Bai M., Xin J., Wang G., Zhang L., Zimmermann R., Yuan Y., and Wu X. 2016. Discovering the k representative skyline over a sliding window. IEEE Transactions on Knowledge and Data Engineering, Vol. 28, No. 8, 2041–2056.

[19] De Matteis T., Di Girolamo S., and Mencagli G. 2016. Continuous skyline queries on multicore architectures. Concurrency and Computation: Practice and Experience, Vol. 28, No. 12, 3503–3522.

[20] Wang Z., Xin J., Ding L., Ba J., and Gao X. 2018. ρ-dominant skyline computation on data stream, IEEE Access, Vol. 6, 53201–53213.

[21] Alami K. and Maabout S. 2020. A framework for multidimensional skyline queries over streaming data, Data and Knowledge Engineering, Vol. 127. 1–21.

[22] Deng Z., Wang Y., Liu T., Dustdar S., Ranjan R., Zomaya A., Liu Y., and Wang L. 2022. Spatial-keyword skyline publish/subscribe query processing over distributed sliding window streaming data, IEEE Transactions on Computers, Vol. 71, No. 10, 2659–2674.

[23] Agrawal, R., Imieliński, T., and Swami, A. 1993. Mining association rules between sets of items in large databases. In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data. ACM, Washington, DC, USA. 207–216.