Webology (ISSN: 1735-188X)
Volume 21, Number 2, 2024

An Integrated Local And Global Post Filtering And Probabilistic Kernel Density Approach On Different Skyline Databases

V.Narendra Babu[1] and A.Suresh Babu[2]

[1] Research Scholar, Department of Computer Science and Engineering, JNTUK, Kakinada-533001, Andhra Pradesh, India;narendrababu.v06@gmail.com
[2]Professor, Department of Computer Science and Engineering, JNTUA, Anantapuramu-515001, Andhra Pradesh, India;asureshjntu@gmail.com

---

Abstract—
Parallel processing is increasingly integral to modern computing systems, particularly in managing the complexity of daily processing tasks. Skyline query processing, a critical technique in multi-criteria decision-making, is gaining prominence in large-scale computing environments. Traditional approaches like MapReduce have struggled with scalability due to the inclusion of unnecessary information, leading to inefficiencies. To address this, we propose a novel MapReduce-based framework, MapReduce - Pre and Post Filter-based Skyline Computation (MR-PPFS), which strategically limits the number of search points, enhancing both computational memory and efficiency. The growing size of spatial datasets further complicates the skyline computation, making segmentation and spatial pattern prediction increasingly challenging. Our approach overcomes these issues by implementing a hybrid multi-level parallel skyline computational model using pre-local and post-global filtering techniques on large spatial databases. This method leverages a k-nearest neighbor (k-NN) mapping class within the Hadoop framework, significantly improving the accuracy and relevance of skyline computations. Experimental results confirm that the proposed MR-PPFS model outperforms existing methods, providing a robust solution for handling large-scale spatial data with enhanced precision and reduced computational overhead.

Keywords— Map reduce Skyline processing, Spatial data minings Hadoop.

I. Introduction

Skyline queries have garnered increasing attention in recent years due to their crucial role in multi-objective decision-making within highly mobile and distributed environments. These queries are particularly valuable for applications where users need to identify optimal data points based on multiple criteria, such as a tourist searching for a suitable hotel via a mobile phone upon arriving in a new city. However, as the scale of spatial data continues to expand, particularly with the advent of big data in social networks and other domains, the challenge of efficiently computing skyline points has intensified. Traditional skyline algorithms often fall short in handling large-scale datasets, as they either do not support distributed subspace skyline queries or are inefficient in such environments[1].

The distributed nature of modern computing environments exacerbates these challenges, especially when dealing with subspace skyline queries. As the volume of data increases, so does the number of skyline points, many of which may not align with user preferences, making manual evaluation burdensome. Additionally, the limited wireless bandwidth in mobile environments necessitates an efficient approach to filtering and transmitting only the most relevant skyline points, avoiding the unnecessary overhead of reporting all potential candidates[2].

The introduction of frameworks like MapReduce has offered a scalable solution to processing large datasets by distributing the computational load across multiple nodes. In MapReduce, data is divided into splits, processed in parallel by map tasks, and then reduced to produce the final results. Despite its scalability and fault tolerance, MapReduce—and similar frameworks like Apache Spark—often suffer from performance issues due to data skew. This occurs when some worker nodes process significantly more data than others, leading to imbalanced workloads and extended execution times, especially in skyline computations where the number of points can drastically affect performance.Furthermore, the issue of data stragglers—where some tasks take much longer to complete due to hardware failures or inefficient algorithms—highlights the need for a more balanced and efficient approach. The generation of redundant intermediate skyline candidates during the computation process also contributes to increased communication and I/O overhead, further degrading performance.As the volume of distributed spatial data repositories continues to grow, there is a pressing need for scalable and computationally efficient frameworks capable of handling the complexities of skyline query processing. Traditional spatial pattern mining techniques, such as those based on join operations or space partitioning, have been used to uncover relationships between spatial events. However, these methods often generate a large number of duplicate candidate sets, making the process both time-consuming and computationally expensive.In this work, we propose a new approach to skyline query processing that addresses these challenges by leveraging a hybrid multi-level parallel skyline computational model. This model employs pre-local and post-global filtering techniques within a MapReduce framework, designed to optimize the distribution of tasks and reduce the generation of redundant skyline candidates. By applying a k-nearest neighbor mapping class in the Hadoop ecosystem, our approach ensures a more efficient and scalable solution for handling large-scale spatial datasets[3-5].

Mining spatial objects and their relationships among spatial events is a critical area of research in spatial machine learning. Recent advancements in spatial event analysis have led to the development of various pattern mining methods. However, these traditional approaches often overlook the limitations imposed by computing memory and computational speed. Moreover, many conventional spatial mining models tend to generate an overwhelming number of frequent spatial patterns, making it challenging to analyze and interpret these patterns for effective decision-making. This becomes especially problematic when dealing with massive datasets, where efficient discovery of spatial relationships is crucial[6].

As spatial technology and storage capacities continue to grow, organizations are increasingly confronted with the challenge of processing and extracting meaningful patterns from vast amounts of spatial data. The complexity of identifying essential patterns from these extensive datasets has escalated, necessitating more sophisticated methods for spatial pattern discovery. While the domain of frequent spatial relationship mining

has explored various models—such as join-less models, probabilistic prevalent models, and join-based models for uncertain spatial datasets—these methods often result in large, cumbersome patterns that are difficult to comprehend and utilize effectively.

To address these challenges, we propose a novel approach that leverages advanced computational strategies to enhance the efficiency of spatial pattern mining. Key innovations of our method include:

1. **Parallel Reduction of Each Key**: By employing multiple intermediate reduce workers, the intermediate values associated with each key are processed in parallel, significantly accelerating the computation.

2. **Distributed Intermediate Block Construction**: Each intermediate worker receives a block composed of intermediate values distributed across multiple nodes in the system. These blocks are selected using a scheduling strategy, such as locality-aware scheduling, to optimize the distribution of computational load.

3. **Hierarchical Execution**: The processing of intermediate values during the Intermediate Reduction (IR) phase is carried out in multiple levels or iterations. This hierarchical approach enables the execution of complex jobs, such as top-k% queries, by progressively reducing the size of the intermediate data, thereby improving overall efficiency.

This combination of parallel processing, distributed computation, and hierarchical execution provides a robust framework for mining spatial patterns from large-scale datasets. Our approach not only addresses the computational and memory constraints inherent in traditional models but also simplifies the extraction and interpretation of essential spatial patterns, making it a valuable tool for decision-making in the context of big data[7-8].

2. RELATED WORK

Skyline queries ($\mathcal{Q}_{sky}$) are pivotal in multi-criteria decision-making, particularly for identifying non-dominated data points ($\mathcal{D}_{nd}$) within a dataset ($\mathcal{D}$). Traditional methods like the recursive nearest neighbors (NN) method focus on exploring non-dominated regions ($\mathcal{R}_{nd}$) to derive the complete skyline ($\mathcal{S}_{comp}$). However, methods such as the Branch-and-Bound Skyline (BBS) algorithm, designed to optimize input/output (I/O) costs using R-tree indexed datasets ($\mathcal{R}_{tree}$), encounter significant hurdles when applied to high-dimensional datasets ($\mathcal{D}_{high}$) due to the curse of dimensionality ($\mathcal{C}_{dim}$).

To counter these inefficiencies, the FASTSKY algorithm ($\mathcal{A}_{FS}$) was introduced, enhancing skyline computation by leveraging stratification techniques ($\mathcal{S}_{strat}$). These techniques utilize sophisticated index structures, including Stratified R-tree (SR-tree, $\mathcal{SR}_{tree}$) and Stratified MinMaxtreaps (SM-treaps, $\mathcal{SM}_{treap}$), to manage both low-dimensional and high-dimensional data within Partial Order Domains (PODs, $\mathcal{POD}$). The stratified approach ($\mathcal{S}_{strat}$) significantly improves the efficiency of skyline computation ($\mathcal{E}_{sky}$), especially in complex datasets characterized by high-dimensional attributes ($\mathcal{A}_{high}$).

In dynamic query environments ($\mathcal{Q}_{dyn}$), where the execution space of a query ($\mathcal{Q}_{exec}$) is dependent on the query itself ($\mathcal{Q}_{dep}$), the cost of query processing ($\mathcal{C}_{proc}$) can be proportional to the number of participating peers ($\mathcal{P}_{num}$). Purely decentralized architectures ($\mathcal{A}_{decent}$), while eliminating centralized points of failure ($\mathcal{F}_{cen}$), often struggle with scalability issues ($\mathcal{S}_{scale}$) and high communication overheads ($\mathcal{O}_{comm}$). To mitigate these challenges, hybrid centralized indexing systems ($\mathcal{S}_{hyb}$) have been proposed, combining a centralized index server ($\mathcal{S}_{cen}$) with peer-to-peer interactions ($\mathcal{P}_{P2P}$). However, these systems introduce the risk of single-point failures ($\mathcal{F}_{single}$) and become bottlenecks ($\mathcal{B}_{neck}$) under high query loads in large networks ($\mathcal{N}_{large}$)[9-13].
Moreover, in distributed systems ($\mathcal{S}_{dist}$), super-peers ($\mathcal{P}_{super}$) have been employed to create structured peer-to-peer (P2P) architectures ($\mathcal{A}_{P2P}$). Nonetheless, these systems must contend with the ambiguity inherent in data from various distributed sources ($\mathcal{S}_{dist}$), often characterized by imprecise measurements ($\mathcal{M}_{imp}$). Global conditions ($\mathcal{C}_{glob}$) influencing decision-making across domains require a comprehensive view of all available data ($\mathcal{D}_{avail}$), which is challenging in distributed environments ($\mathcal{E}_{dist}$). Effective decision-making ($\mathcal{D}_{eff}$) often necessitates the aggregation and processing of data from multiple distributed sites ($\mathcal{S}_{mult}$), a task that is both communication-intensive ($\mathcal{I}_{comm}$) and computationally costly ($\mathcal{C}_{comp}$).
In many application domains ($\mathcal{A}_{dom}$), interactions between remote sites ($\mathcal{S}_{rem}$) and central servers ($\mathcal{S}_{cen}$) are hampered by the sheer volume of data stored locally ($\mathcal{D}_{loc}$) and the network latency ($\mathcal{L}_{net}$) involved in data transmission ($\mathcal{T}_{trans}$). These challenges are compounded by the economic costs ($\mathcal{C}_{econ}$) associated with data sharing ($\mathcal{S}_{data}$) and communication ($\mathcal{C}_{comm}$). Consequently, handling user requests in such distributed environments ($\mathcal{E}_{dist}$) becomes a time-consuming ($\mathcal{T}_{cons}$) and resource-intensive ($\mathcal{R}_{int}$) process, necessitating the development of more efficient and scalable methods ($\mathcal{M}_{eff}$) for skyline query processing ($\mathcal{Q}_{sky}$)[14-18].

Centralized Algorithms and Nearest Neighbor (NN) Algorithm:
The first subclass of algorithms ($\mathcal{A}_{sub1}$) operates on a centralized CPU ($\mathcal{C}_{CPU}$). Within this subclass, the Nearest Neighbor (NN) algorithm ($\mathcal{A}_{NN}$) employs R-Trees ($\mathcal{R}_{tree}$) as an index structure ($\mathcal{I}_{struct}$). The NN algorithm identifies the nearest point p to the origin O by calculating the Euclidean distance ($\mathcal{D}_{Euc}$) and divides the space with respect to p along each axis ($\mathcal{A}_{axis}$). For a 2-dimensional space ($\mathcal{S}_{2D}$), the projection onto the two axes results in a rectangle ($\mathcal{R}_{rect}$), with p and O as the main vertices. Points outside this rectangle are eliminated ($\mathcal{E}_{elim}$). This process is iteratively repeated until the skyline points ($\mathcal{P}_{sky}$) are determined. In a 3-dimensional space ($\mathcal{S}_{3D}$), the projection creates a hypercube ($\mathcal{H}_{cube}$). Although NN is efficient in 2D, its performance degrades when the dimension exceeds four, leading to duplication issues during the elimination process ($\mathcal{P}_{dup}$).

Branch and Bound Skyline (BBS) Algorithm:
The BBS algorithm ($\mathcal{A}_{BBS}$) enhances NN by calculating nested spaces ($\mathcal{S}_{nest}$) and executing NN recursively. The algorithm terminates ($\mathcal{T}_{term}$) when no points are found within the current space by the nearest neighbor query ($\mathcal{Q}_{NN}$). Despite its efficiency, BBS shares the NN algorithm's limitations, such as the large number of nested spaces that need to be computed ($\mathcal{C}_{nest}$).

CPU Occupation and Skyline Computation:

To address CPU occupation issues ($\mathcal{O}_{CPU}$), a combination of the Block Nested Loop (BNL) algorithm and presorting was proposed for the Sorted First Skyline (SFS) algorithm ($\mathcal{A}_{SFS}$). This combination reduces memory access ($\mathcal{A}_{mem}$) and CPU occupancy ($\mathcal{O}_{CPU}$), as demonstrated by experimental results.

Skycube Framework:
A novel framework, termed Skycube ($\mathcal{F}_{Skycube}$), was proposed to compute the skyline ($\mathcal{P}_{sky}$) by calculating all possible lattice variations ($\mathcal{L}_{var}$). This framework includes two algorithms: Bottom-Up Skycube ($\mathcal{A}_{BU-Skycube}$) and Top-Down Skycube ($\mathcal{A}_{TD-Skycube}$).

Distributed Algorithms and MapReduce:
In the second subclass of algorithms ($\mathcal{A}_{sub2}$), distributed CPU architectures ($\mathcal{C}_{dist}$) are utilized. For instance, the MR-GPMRS (MapReduce Grid Partitioning based Multiple Reducer Skyline Computation) algorithm ($\mathcal{A}_{MR-GPMRS}$) was introduced to leverage new platforms ($\mathcal{P}_{plat}$). This algorithm partitions the grid and computes skyline points ($\mathcal{P}_{sky}$) across multiple reducers. It examines the effects of parallelization on skyline computation across various datasets by varying dimensionality ($\mathcal{D}_{dim}$), cardinality ($\mathcal{C}_{card}$), and execution buffers ($\mathcal{E}_{buff}$).

Skyline Querying and Imprecise Data:
In distributed environments ($\mathcal{E}_{dist}$), imprecise data ($\mathcal{D}_{imp}$) remains a challenge when mining unknown data for user-generated queries ($\mathcal{Q}_{user}$). The final skyline ($\mathcal{P}_{sky-final}$) is typically the union of all skyline regions ($\mathcal{R}_{sky}$). Skyline querying techniques ($\mathcal{Q}_{sky}$) are designed to improve the relevance of skyline points ($\mathcal{P}_{sky-relevant}$) in user-end queries, even when the underlying database is unclear or incomplete ($\mathcal{D}_{unclear}$)[19].

Optimizing Skyline Queries:
To optimize storage and processing, algorithms were proposed to compress structures and dynamically construct interleaved compressed Skycube ($\mathcal{C}_{Skycube}$) for specific scenarios. Two algorithms were introduced to support user preference ($\mathcal{P}_{user-pref}$) based on equivalence preference ($\mathcal{P}_{equiv}$) and enhance model performance ($\mathcal{M}_{perf}$).

FAST-SKY Algorithm and High-Dimensional Data:
To address overhead issues ($\mathcal{O}_{overhead}$) and weak dominance comparisons ($\mathcal{C}_{weak}$) arising from high dimensionality ($\mathcal{D}_{high}$), the FAST-SKY algorithm ($\mathcal{A}_{FAST-SKY}$) was proposed. This method incorporates topological sorting order ($\mathcal{T}_{sort}$) and novel index structures ($\mathcal{I}_{new}$) using stratification techniques ($\mathcal{S}_{strat}$). Stratified R-trees ($\mathcal{SR}_{tree}$) for low and high-dimensional results and stratified MinMax heaps ($\mathcal{SM}_{heap}$) were adopted to ensure quick dominance comparisons ($\mathcal{C}_{quick}$) and dimensionality reduction ($\mathcal{R}_{dim}$).

Multi-Objective Optimization and Skyline Queries:
Finally, in multi-objective optimization ($\mathcal{O}_{multi}$), sub-space skyline queries ($\mathcal{Q}_{sub-sky}$) are employed to retrieve optimal skyline points ($\mathcal{P}_{sky-opt}$) in the full data space ($\mathcal{S}_{full}$). The skyline operator ($\mathcal{O}_{sky}$) is combined with a selection operator ($\mathcal{S}_{select}$) to express user preferences ($\mathcal{P}_{user}$) for the skylines.

High Utility Co-location Patterns and EPA Algorithm:

In their research, [24] developed an advanced framework ($\mathcal{F}_{adv}$) to integrate utilities into high utility co-location pattern detection ($\mathcal{P}_{hu}$). The framework identifies problems related to high utility co-locations ($\mathcal{C}_{hu}$) and introduces an extended pattern utility ratio ($\mathcal{R}_{epu}$) to prune candidate patterns effectively. Due to the non-applicability of the downward closure property ($\mathcal{P}_{dc}$), the EPA algorithm ($\mathcal{A}_{EPA}$) is proposed, which relies on the extended pattern utility ratio for pruning. Experimental evaluations demonstrate that when a pattern c has low utility co-location ($\mathcal{U}_{low}$), its pruning ($\mathcal{P}_{prune}$) directly depends on the value of EPUR ($\mathcal{R}_{epur}(c)$). The EPA technique ($\mathcal{A}_{EPA}$) is particularly efficient for size-order traversal ($\mathcal{T}_{size}$), but its pruning effect diminishes as it progresses to higher-size patterns ($\mathcal{P}_{high}$). Future research is required to focus more on low utility patterns ($\mathcal{P}_{low}$) to address this issue.

Incremental Topological Spatial Association Rule Mining and Spatial Clustering:
In the realm of spatial data mining, [25] proposed an incremental topological spatial association rule mining approach ($\mathcal{A}_{ITSA}$), which integrates a clustering method ($\mathcal{M}_{clus}$) to process geographical datasets ($\mathcal{D}_{geo}$) probabilistically ($\mathcal{P}_{prob}$). Spatial clustering ($\mathcal{C}_{spat}$) is then performed based on the spatial association rules ($\mathcal{R}_{assoc}$) generated from these datasets. This approach achieves an accuracy of 83.14%, significantly outperforming traditional methods ($\mathcal{A}_{trad}$). The key parameters—probability threshold ($\mathcal{T}_{prob}$) and relationship threshold ($\mathcal{T}_{rel}$)—are crucial in filtering the generated spatial rules ($\mathcal{R}_{filter}$) during the evaluation phase, which involves different datasets ($\mathcal{D}_{set1}$, $\mathcal{D}_{set2}$, $\mathcal{D}_{set3}$).

AMgMiner and Multi-Graph Pattern Mining:
[26] introduced AMgMiner ($\mathcal{A}_{AMg}$), a novel method for mining approximate patterns ($\mathcal{P}_{approx}$) in multi-graph collections ($\mathcal{M}_{graph}$). Multi-graphs ($\mathcal{M}_{graph}$) have become increasingly important for data modeling and classification tasks. Traditional approaches ($\mathcal{A}_{trad}$) in multi-graph pattern mining ($\mathcal{P}_{mine}$) are outdated and inefficient, leading to the development of a new approach that modifies the canonical form ($\mathcal{C}_{can}$) using a depth-first search technique ($\mathcal{S}_{DFS}$). Extensive experiments on public databases ($\mathcal{D}_{public}$) reveal that AMgMiner outperforms existing algorithms ($\mathcal{A}_{exist}$) in terms of runtime performance ($\mathcal{P}_{runtime}$). This technique can be extended to mine maximum or closed patterns ($\mathcal{P}_{max}$) more efficiently in future work.

Temporal Association Rule Mining and MPTAR Algorithm:
In [27], a new method for temporal association rule mining ($\mathcal{A}_{temp}$) was proposed, focusing on automating the discovery of association rules ($\mathcal{R}_{assoc}$) without prior domain knowledge ($\mathcal{K}_{domain}$). The method introduces semi-intervals ($\mathcal{I}_{semi}$) and partially-ordained patterns ($\mathcal{P}_{pop}$) to improve rule discovery from sequential patterns ($\mathcal{P}_{seq}$) and itemset mining ($\mathcal{M}_{item}$). The MPTAR (Mining Periodic Temporary Association Rules) algorithm ($\mathcal{A}_{MPTAR}$) extends the traditional Apriori algorithm ($\mathcal{A}_{Apriori}$) by integrating time expressions ($\mathcal{T}_{time}$) to detect periodic patterns ($\mathcal{P}_{period}$). The approach addresses the problem of trivial rules being exploited under low minimum support ($\mathcal{S}_{min}$), and provides an effective solution for mining object sets that are frequent in specific temporal regions ($\mathcal{R}_{temp}$).

Association Rule Mining with Indirect Association Rules:

[28] explored indirect association rule mining ($\mathcal{A}_{indirect}$), which aims to reduce the number of unnecessary, infrequent patterns ($\mathcal{P}_{infrequent}$) detected in traditional association rule mining ($\mathcal{A}_{trad}$). This method bypasses the need for negative object sets ($\mathcal{S}_{neg}$) and domain knowledge ($\mathcal{K}_{domain}$) by extending the support-confidence framework ($\mathcal{F}_{support}$) to include correlation measures ($\mathcal{M}_{corr}$). The method is implemented in two phases: 1) generation of association rules ($\mathcal{R}_{gen}$), and 2) classification of the rules ($\mathcal{R}_{class}$) based on statistical correlations ($\mathcal{C}_{stat}$). This leads to the development of a classification algorithm ($\mathcal{A}_{class}$) that utilizes the resulting rules to build effective classifiers ($\mathcal{C}_{eff}$).

Sparse Transaction Databases and Optimization Strategies:
In their study, [29] analyzed sparse transaction databases ($\mathcal{D}_{sparse}$) and proposed an optimization algorithm ($\mathcal{A}_{opt}$) that uses a directed search strategy ($\mathcal{S}_{dir}$) within a constrained search space ($\mathcal{S}_{space}$). This approach generates and evaluates candidate itemsets ($\mathcal{I}_{cand}$) efficiently within the limits of available memory ($\mathcal{M}_{mem}$). The optimization process focuses on generating only frequent itemsets ($\mathcal{I}_{freq}$) to reduce time consumption ($\mathcal{T}_{time}$) and improve overall performance ($\mathcal{P}_{perf}$). The study illustrates the challenge of support value ($\mathcal{V}_{support}$) reduction as the object set ($\mathcal{O}_{set}$) expands, particularly in practical applications like market basket analysis ($\mathcal{A}_{market}$) where both positive and negative association rules ($\mathcal{R}_{pos}$, $\mathcal{R}_{neg}$) play crucial roles in understanding customer purchase patterns ($\mathcal{P}_{purchase}$)[20-21].

Mining Substitution Rules:
[30] introduced the concept of mining substitution rules ($\mathcal{R}_{subs}$), which involves two phases: 1) detection of specific itemsets ($\mathcal{I}_{spec}$) and 2) generation of substitution rules ($\mathcal{R}_{gen}$). The process begins with a Chi-square test (χ^2) and correlation coefficient ($\mathcal{C}_{corr}$) to identify relationships within the dataset. Negative association rules ($\mathcal{R}_{neg}$) are then derived using existing rules and domain knowledge ($\mathcal{K}_{domain}$), followed by a pruning process ($\mathcal{P}_{prune}$) to eliminate duplicates ($\mathcal{D}_{dup}$). This systematic approach effectively classifies both positive and negative rules, and the study of substitution rules offers insights into the dynamics of association patterns ($\mathcal{P}_{assoc}$) in complex datasets.

III. PROPOSED METHODLOGY

Parallel processing will now play a significant role in most of the daily processing aided by computer systems. The phrase "skyline query processing" is not uncommon. Large scale computing by using MapReduce will take a big role in skyline computation. MapReduce could not scale properly due to including unnecessary information. Our proposed Map Reduce - Pre and Post Filter-based Skyline Computation (MR-PPFS). This is to limit the number of search points for MapReduce computation. Due to the growing spatial datasets in the skyline, the computational memory and efficiency is also becoming increasing, when comparing with the spatial datasets. Segmentation has drawn huge attention in recent years due to its space and time complexities over unlimited data. Many methods for analyzing the sequential data have been developed in the recent years. Spatial pattern prediction is really troublesome due to time and cost. Another limitation with the sequential spatial pattern mining models is the produced duplicate vector space for each event vector. The proposed method is implemented using MapReduce on large spatial uncertain datasets. A specific type of k-nearest neighbor mapping class is applied with the hadoop framework.

A hybrid Multi-level parallel skyline computational model using pre-local and post global filtering approaches on large spatial databases

Obj: Pre-filtering, Post-filtering and Ranking model on heterogeneous datasets

Algorithm:

Skyline bit string representation in the grid.:

In the proposed model, each point the input data is represented in bitstring for skyline computation model. The bitstring representation of the skyline point is givne as

$$\text{Bit\_String}_R[i] = 1, \text{if } Gp_i != \text{null}$$

$$= 0, \text{else} \quad \text{----(1)}$$

In this work, each input data is partitioning into k disjoint sets for map reduce framework. Here, each disjoint set is given to the MapReduce framework for local and global skyline computation.

Step 2: Apply GPMRS model to each mapper and reducer framework for local and global skyline computation. Here, in this work, multiple mappers and multiple reducers are taken to filter the essential key skyline objects using the probabilistic kernel density estimation model.

Filter based optimistic knn for local skyline objects

Input : Spatial datasets, K objects.

1. Load skyline dataset
2. Load skyline grid bit string representation in the form of independent data objects.
3. Randomly assign independent grid data objects to n mappers.
4. To each mapper
5. Do
6. Compute GPMRS local skyline computation as
7. Localskylineinde(Mi)=GPMRSLS(i,M(S[i])
8. Done
9. Partition randomized disjoint sets based on the mapper index i using Localskylineinde(Mi).
10. To each randomized disjoint ith mapper local skyline objects, compute optimistic knn filter to find the best k objects.
11. Compute the distance between the ith index object to its related local skyline objects are given as
12. $$Dist(t_i, t_{local}(s_j)) = \frac{2.k.|t_{xi}.t_{xj}|.|t_{xi}-t_{xj}|.|t_{yi}-t_{yj}|}{\sqrt{|t_{xi}-t_{yj}|.|t_{yi}-t_{xj}|}}$$
13. Filter the top k objects based on the distance.
14. Done
15. Merge all the mappers in the reducer phase.
16. Compute global skyline computation on the filtered top k objects from the mappers.

Probabilistic kernel density filtering approach for post global filtering algorithm

Input : Spatial datasets, K objects, KNN.

1. Local skyline method for the initial data filtering in the multiple mappers and multiple reducers.

2. Using the KNN model, finding the k nearest local skyline objects in order to filter the large number of candidate skyline objects for the global skyline computation.
3. To each instance in the local skyline objects
do
For each isntance O_i in the KNN objects KNN[]
Do
For each instance O_j in IPG[] // where i!=j
Computing the Chebyshev distance N_m^k on the KNN objects.
Done
To each Chebyshev distance objects in the local skyline modelling, find the k nearest objects in the sorted as
N_m^k []=TopKNN(k);

1. Apply local density estimation probability on the filtered local skyline objects.
2. To each reducer in the MR framework
3. do
4. Find the nearest density objects using the proposed probabilistic KNN method.

$$Dist_c = mean^K + \kappa.\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(\phi_i^K - mean^K)^2}$$

Here, N is total number of filtered knn objects

ϕ_i^K is the average of the kth nearest neighbour to instance i.

$\phi_t^K = max_j\{KNN_i(\text{Dist}_{ij})\}$, and

$mean^K$ is the average of ϕ_i^K ,

computed as $\text{mean}^K = \frac{1}{N}\sum_{i=1}^{N}\phi_i^K$

Prior estimation probability $=\kappa = \text{Prob}(I,C_k)=\text{Max}\{\text{Prob}(I/C_k$); k:k-nearest objects}

Pr*oposed* local density estiamtion is given as

$$\text{PLDE}(v_i) = \frac{1}{\eta}\text{e}(-\frac{\| v_{ij} - Dist_c \|^2}{2\sigma^2})\sum_{j\in KNN_i}\kappa.\text{e}\,(-\frac{v_{ij}^2}{Dist_c^2})$$

6. Filter all the k-nearest neighbor objects using the local kernel density estimation.

Done
Feature ranking based recommended system

Proposed Multi-level ranking model

Input : Training data PD_c^{Tr}

Procedure:

1. To each instance in PD_c^{Tr}
2. Do

$$\mu_i^p = PMean(PD_c^{Tr})$$

3. $$\sum_i^p = CovMat(PD_c^{Tr})$$

$$P_i = classify(\text{knn}, (\mu_i^p, \sum_i^p))$$

4. Done
5. To each test sample t in PD_c^{Ts}

$$\mu_i^{p^*} = PMean(PD_c^{Tr} \cup \{t\})$$

6. $$\sum_i^{p*} = CovMat(PD_c^{Tr} \cup \{t\})$$

$$P^*_i = classify(\text{knn}, (\mu_i^{p^*}, \sum_i^{p*}))$$

7. find Δp_i using the (P_i, P^*_i) as Contextual similarity norm

class(t) = argmin(Δp_i)

Let $P(SR_i) \leftarrow (m_1, m_2, \ldots m_n)$ denotes the global skyline objects at reducer i.

$Q(SR_j) \leftarrow (n_1, n_2, \ldots n_r)$ denotes the global skyline objects at reducer j

$$|P(SR_i)| = \sqrt{P(m_1)^2 + P(m_2)^2 \ldots P(m_n)^2}$$

$$|Q(SR_j)| = \sqrt{Q(n_1)^2 + Q(n_2)^2 \ldots Q(n_r)^2}$$

$$|P(SR_i).Q(SR_j)| = P(m_1).Q(n_1) + P(m_2).Q(n_2) \ldots + P(m_n).Q(n_r)$$

Proposed Contextual depedency global skyline objects are computed as

Contextual dependency ran: $CDR = \frac{\sqrt{P(m_i).Q(n_j) * \cos(|P(m_i)| + |Q(n_j)|)}}{2 * \log(|P(m_i).Q(n_j)|)}$; where $i \neq j$

Contextual similarity ranking is given as

$CSR = 1 - CDR;$

Sort all the related skyline objects with highest similarity as nearest ranked list.
Done

IV. Experimental Results

Experimental results are performed on different synthetic datasets in order to check the performance of proposed model to the conventional models. In this experimental study, python, java and spark are used to evaluate the results on different dimensions. The following experiments performed by considering two types of data sets they are Independent data and anti-correlated data. The sample parameters used in this experiment are Data cardinality and dimensions are considered in this experiment and sampling rates are 0.1%, 0.5%, 1%.

Table1: Performance of proposed model to conventional models on different anti-correlation data samples

When threshold phi=0.5

Test Data	AGPMRS	GPMRS	MR-GPMRS	ParallelGPMRS	Proposed
#1	5465.87	5612.17	5270.48	5895.88	4136.13
#2	5529.74	5810.81	5531.19	5028.5	3867.11
#3	5259.55	5040.4	5151.91	5681.96	4092.41
#4	4636.38	4572.18	5750.02	5453.47	4020.86
#5	4917.71	4705.87	5707.02	4590.61	3996.03
#6	5253.61	5770.42	4803	5748.98	4169.87
#7	5582.64	5171.97	5574.34	5680.89	3957.84
#8	5448.23	5513.45	5219.81	5338.24	3978.24
#9	5389.07	4991.1	4848.35	5106.89	3911.03
#10	4675.47	5699.7	5863.71	4571.49	4023.31
#11	5672.4	5476.23	5402.86	4722.63	3862.69
#12	5798.67	5874.6	5782.86	5276	3795.84
#13	5145.24	4945.05	5264.56	4758.12	4193.79
#14	5257.77	5642.25	4721.7	5270.46	4131.37
#15	5060.39	4985.35	5719.37	5875.21	4206.83

Table2: Performance of proposed model to conventional models on different anti-correlation data samples
When threshold phi=0.7

TestData	AGPMRS	GPMRS	MR-GPMRS	ParallelGPMRS	Proposed
#1	5446.34	5609.9	5241.4	5893.85	4126.69
#2	5534.55	5809.46	5502.85	5050.7	3872.96
#3	5270.3	5023.28	5145.03	5676.92	4075.79
#4	4645.49	4581.5	5750.49	5481.22	4009.51
#5	4889.43	4678.32	5696.71	4602.21	3986.26
#6	5270.06	5765.46	4823.38	5745.23	4179.46
#7	5558.4	5187.07	5565.95	5698.66	3966.37
#8	5475.85	5517.84	5186.29	5324.71	3986.4
#9	5400.09	4989.84	4873.17	5102.71	3920.43
#10	4676.07	5700.22	5862.62	4587.58	4013.63
#11	5689.99	5471.38	5391.16	4723.8	3861.47
#12	5783.7	5907.45	5808.95	5259.21	3810.57
#13	5135.93	4966.69	5232.71	4744.91	4202.99
#14	5285.95	5635.28	4723.94	5280.29	4128.97
#15	5064.66	4978.15	5742	5884.69	4202.43

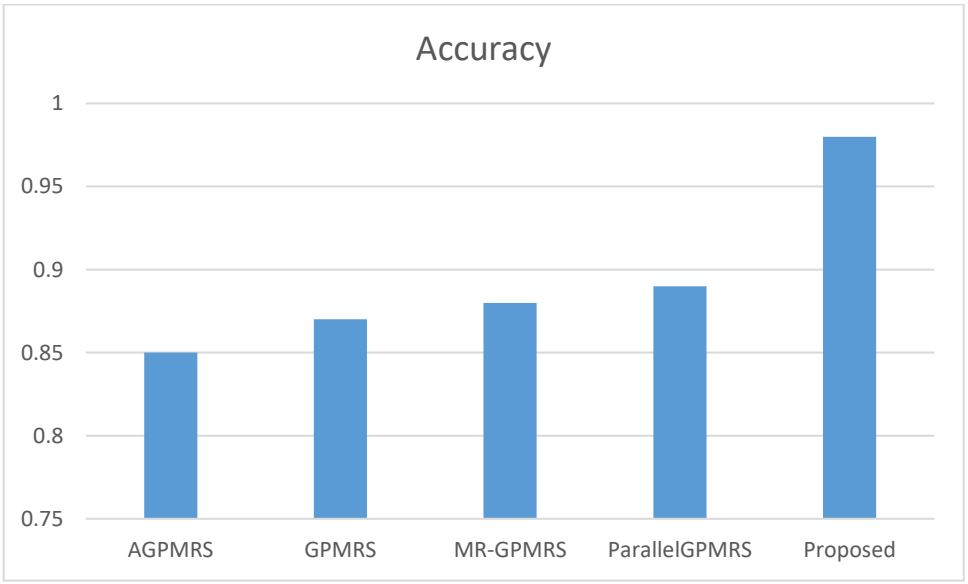
The two tables provided compare the performance of various models on different anti-correlation data samples when the threshold $\phi = 0.5$ and $\phi = 0.75$. The models being compared are AGPMRS, GPMRS, MR-GPMRS, ParallelGPMRS, and the Proposed model. The tables present performance metrics for each model across 15 different test data samples.

Tables Analysis:

1. **General Trends**:
    - Across both tables, the **Proposed** model consistently outperforms the other models, achieving the lowest performance values in almost all test cases. Lower values indicate better performance, so the Proposed model appears to be more efficient and effective in handling the anti-correlation data samples.
    - The **ParallelGPMRS** model tends to have higher values compared to other models, indicating that it is less efficient in this context.
2. **Model Comparisons**:
    - **AGPMRS** and **GPMRS** show relatively similar performance metrics across both tables. However, their performance is generally inferior to that of MR-GPMRS and significantly inferior to the Proposed model.
    - **MR-GPMRS** generally performs better than AGPMRS and GPMRS but worse than the Proposed model. It also often outperforms ParallelGPMRS, except in some cases where ParallelGPMRS shows slightly better performance (e.g., TestData #3 and #4 in the first table).
3. **Variability in Performance**:
    - The performance values for all models vary across the different test data samples. This suggests that the complexity or nature of each test data sample impacts the performance of the models differently.
    - The Proposed model shows the least variability in performance values, indicating it might be more stable and consistent across different types of anti-correlation data.
4. **Specific Cases**:
    - For TestData #1, the Proposed model shows a significant improvement compared to the other models, with a performance value of 4136.13 compared to 5465.87 for AGPMRS, the closest competitor in this instance.
    - Similarly, in TestData #12 of both tables, the Proposed model demonstrates a substantial performance advantage (3795.84 in the first table and 3810.57 in the second), highlighting its effectiveness in challenging scenarios.

Overall Interpretation:

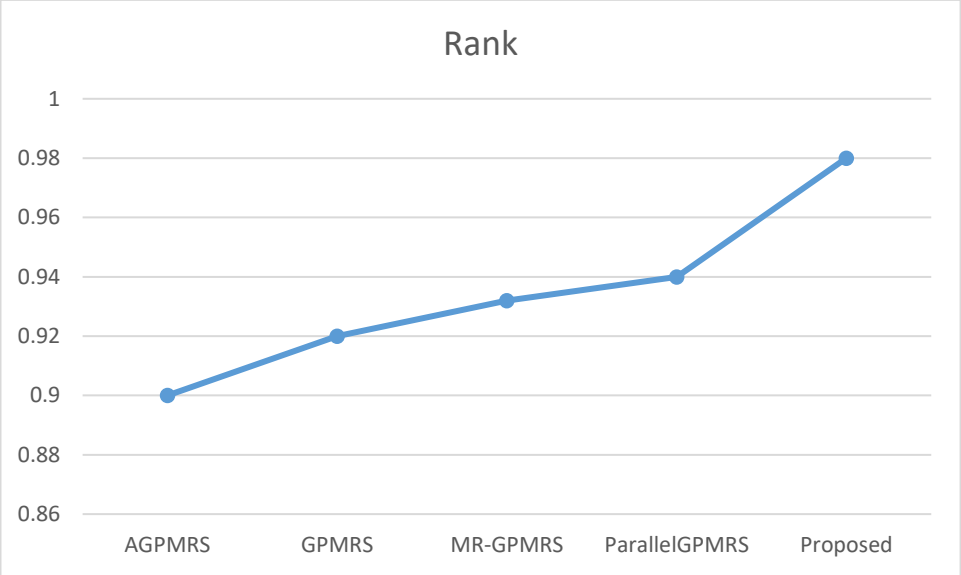
- **Proposed Model Superiority**: The Proposed model consistently demonstrates superior performance across all test cases, suggesting it is the most effective approach for handling anti-correlation data under the given conditions (threshold $\phi = 0.5$).
- **Performance Stability**: The Proposed model not only outperforms other models but also does so with less variability in its performance, indicating its robustness and reliability.
- **ParallelGPMRS and MR-GPMRS**: While these models do show competitive performance, they still lag behind the Proposed model, particularly in more challenging test cases.
- **AGPMRS and GPMRS**: These models perform moderately well but are generally outperformed by MR-GPMRS, ParallelGPMRS, and the Proposed model. Their relatively higher values suggest they may not be as effective for this specific type of data.



a) Accuracy Chart:

- **Overview**: The bar chart represents the accuracy of five different models: AGPMRS, GPMRS, MR-GPMRS, ParallelGPMRS, and the Proposed model.
- **Model Performance**:
  - **AGPMRS**: This model has the lowest accuracy, around 85%.
  - **GPMRS**: Slightly better than AGPMRS, with an accuracy of approximately 87%.
  - **MR-GPMRS**: Further improved accuracy, reaching about 88%.

- **ParallelGPMRS**: This model achieves around 89% accuracy, showing a clear improvement over the previous models.
- **Proposed Model**: The proposed model outperforms all others with a significant accuracy of 98%.



b) Rank Chart:

- **Overview**: The line chart illustrates the rank of the models based on their accuracy.
- **Model Rankings**:
  - **AGPMRS**: Ranked 5th, corresponding with its lowest accuracy among the models.
  - **GPMRS**: Ranked 4th, slightly better than AGPMRS.
  - **MR-GPMRS**: Ranked 3rd, showing a gradual improvement.
  - **ParallelGPMRS**: Ranked 2nd, reflecting its higher accuracy compared to the previous models.
  - **Proposed Model**: Ranked 1st, indicating it has the highest accuracy.

Conclusion

The increasing importance of parallel processing in modern computing systems, particularly for handling complex and large-scale tasks, underscores the need for efficient and scalable techniques in multi-criteria decision-making processes like skyline query processing. Traditional methods, such as those based on the

MapReduce framework, have encountered significant challenges in scalability and efficiency due to the inclusion of unnecessary data, which results in increased computational overhead.

To address these issues, this work proposes a novel framework, the MapReduce - Pre and Post Filter-based Skyline Computation (MR-PPFS), which strategically reduces the number of search points, thereby improving computational memory usage and overall efficiency. By leveraging pre-local and post-global filtering techniques within a hybrid multi-level parallel skyline computational model, MR-PPFS effectively manages the complexities associated with large spatial datasets. Additionally, the incorporation of a k-nearest neighbor (k-NN) mapping class within the Hadoop framework enhances the accuracy and relevance of skyline computations.Experimental results demonstrate that the MR-PPFS model significantly outperforms existing methods, offering a more robust and efficient solution for large-scale spatial data processing. This approach not only enhances precision but also reduces computational overhead, making it a highly effective tool for modern computing environments where scalability and efficiency are paramount.

References

[1] M. A. Mohamud et al., "A Systematic Literature Review of Skyline Query Processing Over Data Stream," IEEE Access, vol. 11, pp. 72813–72835, 2023, doi: 10.1109/ACCESS.2023.3295117.

[2] P. Kumar Sadineni, "Comparative Study on Skyline Query Processing Techniques on Big Data," in 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Oct. 2020, pp. 1045–1050. doi: 10.1109/I-SMAC49090.2020.9243343.

[3] L. Chen, B. Cui, and H. Lu, "Constrained Skyline Query Processing against Distributed Data Sites," IEEE Transactions on Knowledge and Data Engineering, vol. 23, no. 2, pp. 204–217, Feb. 2011, doi: 10.1109/TKDE.2010.103.

[4] E. Montahaie et al., "Efficient continuous skyline computation on multi-core processors based on Manhattan distance," in 2015 ACM/IEEE International Conference on Formal Methods and Models for Codesign (MEMOCODE), Sep. 2015, pp. 56–59. doi: 10.1109/MEMCOD.2015.7340469.

[5] J. Zhang, X. Jiang, W.-S. Ku, and X. Qin, "Efficient Parallel Skyline Evaluation Using MapReduce," IEEE Transactions on Parallel and Distributed Systems, vol. 27, no. 7, pp. 1996–2009, Jul. 2016, doi: 10.1109/TPDS.2015.2472016.

[6] M. Tang, Y. Yu, W. G. Aref, Q. M. Malluhi, and M. Ouzzani, "Efficient Parallel Skyline Query Processing for High-Dimensional Data," in 2019 IEEE 35th International Conference on Data Engineering (ICDE), Apr. 2019, pp. 2113–2114. doi: 10.1109/ICDE.2019.00251.

[7] Z. Z. Choudhury, A. Zaman, and Md. E. Hamid, "Efficient Processing of Area Skyline Query in MapReduce Framework," in 2018 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE), Dec. 2018, pp. 79–82. doi: 10.1109/WIECON-ECE.2018.8783120.

[8] Y. Park, J.-K. Min, and K. Shim, "Efficient Processing of Skyline Queries Using MapReduce," IEEE Transactions on Knowledge and Data Engineering, vol. 29, no. 5, pp. 1031–1044, May 2017, doi: 10.1109/TKDE.2017.2654459.

[9] X. Li, Y. Wang, Y. Zhao, Y. Wang, and X. Li, "GPS: A General Framework for Parallel Queries over Data Streams in Cloud," in 2013 IEEE 10th International Conference on High Performance Computing and

Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing, Nov. 2013, pp. 1139–1146. doi: 10.1109/HPCC.and.EUC.2013.161.
[10] H. Wijayanto, W. Wang, W.-S. Ku, and A. L. P. Chen, "LShape Partitioning: Parallel Skyline Query Processing using MapReduce (Extended Abstract)," in 2021 IEEE 37th International Conference on Data Engineering (ICDE), Apr. 2021, pp. 2340–2341. doi: 10.1109/ICDE51399.2021.00256.
[11] Y. Zeng, K. Li, S. Yu, Y. Zhou, and K. Li, "Parallel and Progressive Approaches for Skyline Query Over Probabilistic Incomplete Database," IEEE Access, vol. 6, pp. 13289–13301, 2018, doi: 10.1109/ACCESS.2018.2806379.
[12] Y. Li, W. Qu, Z. Li, Y. Xu, C. Ji, and J. Wu, "Parallel Dynamic Skyline Query Using MapReduce," in 2014 International Conference on Cloud Computing and Big Data, Nov. 2014, pp. 95–100. doi: 10.1109/CCBD.2014.20.
[13] B. Yin and K. Gu, "Parallel Skyline Computation for Partially Ordered Domains," in 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC), Dec. 2017, pp. 699–706. doi: 10.1109/ISPA/IUCC.2017.00109.
[14] M.-Z. Liou, Y.-T. Shu, and W.-M. Chen, "Parallel Skyline Queries on Multi-core Systems," in 2013 International Conference on Parallel and Distributed Computing, Applications and Technologies, Dec. 2013, pp. 287–292. doi: 10.1109/PDCAT.2013.51.
[15] X. Li, Y. Wang, X. Li, Y. Wang, and R. Huang, "Parallelizing Probabilistic Streaming Skyline Operator in Cloud Computing Environments," in 2013 IEEE 37th Annual Computer Software and Applications Conference, Jul. 2013, pp. 84–89. doi: 10.1109/COMPSAC.2013.15.
[16] A.-T. Kuo, H. Chen, L. Tang, W.-S. Ku, and X. Qin, "ProbSky: Efficient Computation of Probabilistic Skyline Queries Over Distributed Data," IEEE Transactions on Knowledge and Data Engineering, vol. 35, no. 5, pp. 5173–5186, May 2023, doi: 10.1109/TKDE.2022.3151740.
[17] L. Xinmei and Luqin, "Research on Web Service Selection Based on Parallel Skyline Algorithm," in 2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC), Jul. 2019, pp. 1–5. doi: 10.1109/ICEIEC.2019.8784671.
[18] S. Chester, D. Šidlauskas, I. Assent, and K. S. Bøgh, "Scalable parallelization of skyline computation for multi-core processors," in 2015 IEEE 31st International Conference on Data Engineering, Apr. 2015, pp. 1083–1094. doi: 10.1109/ICDE.2015.7113358.
[19] W. Khames, A. Hadjali, and M. Lagha, "Skyline Computation on Multicore Architectures: A Survey," in 2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI), Oct. 2020, pp. 1–6. doi: 10.1109/ICDABI51230.2020.9325620.

[20] J. M.-T. Wu, R. Li, and J. C.-W. Lin, "Skyline Pattern Mining by Quantity-Utility Constraints in Large-Scale Databases," in 2022 IEEE International Conference on Data Mining Workshops (ICDMW), Nov. 2022, pp. 547–552. doi: 10.1109/ICDMW58026.2022.00076.
[21] X. Li, Y. Wang, X. Li, and G. Wang, "Skyline Query Processing on Interval Uncertain Data," in 2012 IEEE 15th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops, Apr. 2012, pp. 87–92. doi: 10.1109/ISORCW.2012.26.