International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V71I6P218

Volume 71 Issue 6, 152-168, June 2023

Original Article

Optimization of Spatio Temporal Aggregate Queries using Semantic Load shedding and Shared Cluster Based Execution

A. Nishad[1], K. P. Noufal[2], N. Rajesh[3], Deepa Mary Mathews[4]

[1]*Department of Higher Secondary Education, Kerala, India*
[2]*Department of Computer Science, NAM College Kallikkandy, Kerala, India*
[3]*Department of Computer Applications SAS SNDP Yogam College, Konni, Kerala, India*
[4]*Department of Computer Applications, Federal Institute of Science and Technology, Kerala, India*

[1]*Corresponding Author : an.nishad@gmail.com*

Received: 20 February 2023 Revised: 07 May 2023 Accepted: 23 May 2023 Published: 25 June 2023

Abstract *- The problem of moving object data modelling secured a great deal of attention due to the wide acceptance of context-based computing frameworks and related applications. This digital revolution has geared momentum in spatio-temporal data mining and continuous query processing research. Efficient approaches in representation, storage, processing and querying of spatio-temporal trajectory are the need of time for providing cost-effective solutions to many problems, especially in transportation systems. The mobility data requires additional considerations in storage and processing due to its dynamic nature. Besides the explicit geographical and temporal data, the trajectory of a moving object contains crucial information about the object's movement and behavior. These semantic features are the factors that connect the meaning and objective of the move. The findings arrived at based on explicit data can give better insights into the moving entity. Aggregation is another effective approach to make use of in this scenario in order to get the collective behaviour of moving objects or the region of travel. In this paper, we propose methods for the convenient representation and effective processing of moving entities. The moving object aggregate queries are grouped into two classes static spatiotemporal aggregate query and continuous spatiotemporal aggregate query. Here, we utilize the semantic-based load shedding over the moving object clusters to reduce the computational overhead. Shared cluster-based execution is incorporated for the effective computation of moving object aggregate queries. Different data structures for representing moving object clusters are also introduced in this paper. Various evaluations are provided to showcase the effectiveness of the approach.*

Keywords *- Data load shedding, Moving object queries, Semantic processing, Spatio temporal knowledge data extraction, Trajectory processing.*

1. Introduction

The world is witnessing an increased diffusion of context-sensing devices, making continuous monitoring of objects along their travel paths a prerequisite for spatiotemporal knowledge extraction. Moving object data analysis has diverse applications, creating a wider perspective on mobility research problems. Location-Based Services (LBS) offer several interesting and challenging application areas, such as the location-based administration of products, services, devices, and people, which are major drivers of m-commerce (mobile commerce). For example, analyzing the travel patterns of different categories of vehicles over a transportation network can provide insights into potential bottlenecks, alternative route suggestions, and the identification of emergency evacuation routes. In the field of tourism, spatiotemporal analysis can be used to prepare travel itineraries, identify popular destinations based on user priority, and suggest trendy travel sequences. Patient path tracing is another relevant area that provides scope for various mitigation strategies [1] Spatio; temporal analysis also finds applications in many areas, such as intelligent manufacturing and animal migration analysis etc.

The moving objects generate a continuous stream of data, making it difficult for a traditional database to process. Heavy computational resources are required to process the voluminous and high-velocity mobility data. Meanwhile, it is evident that in order to get the abstract idea of moving objects, complete details of them are not required. In the spatiotemporal environment, processing individual tuples is not feasible to retrieve the aggregate values for the following

reasons [2]. First, personal attributes (registration no. of vehicle, personal id etc.) are not tracked due to the policies of privacy protection standards. Second, individual records may not always be available while following the data. Third, traces are generated more frequently according to the capacity of the sensors. Hence storage of all individual records is not feasible. For example, to analyse the vehicular density in a popular area, the non-spatial attributes of individual vehicles (registration number, colour etc.) are irrelevant. The straightforward technique for the data analysis of moving objects is the periodical clustering of spatio-temporal variables. While adopting clustering in this context, many additional measures must be taken. The selection of clustering intervals is crucial. If the interval is short, the process becomes expensive and turns down the advantage of periodical clustering. For long period intervals, the chances are there for the loss of vital information[3].

The blooming of context-aware systems has led to the introduction of mobile database systems, which paved the path for mobile query processing. With reference to the context, queries can be either data queries or location-based queries[3]. If the query result is evaluated only once, such queries are termed static queries. However, if the results are evaluated until a termination condition is reached, they are referred to as continuous queries. The termination can be any spatial or temporal constraint in a mobile environment. Continuous queries have created a wider perspective on mobility research problems. For example, consider a practical scenario a person in a moving car issues a query seeking police assistance, "Give the details of police vans that are moving towards the outer ring road, updated every 10 minutes." In this case, both the source and destination entities are moving in a specified direction'; also, after issuing the query, it has to be evaluated periodically with respect to changing spatial and temporal contexts. Particularly, moving object queries are SQL-Like queries required to fulfil spatio-temporal attributes. Successive processing of all records to answer spatio-temporal queries are expensive and unproductive. According to the mode of mobility, a moving object query can be in any of the three scenarios: moving queries on stationery objects, stationery queries on moving objects and moving queries on moving objects. While processing each query category, one needs to consider the mobility characteristics of different environments, the object in motion and the queried domain. We find inspiration from the SCUBA method suggested by Nehme et al. in [4] for managing moving object queries.

Aggregation is a useful concept that can be applied to spatiotemporal data. It aims to represent the collective behavior of a group while preserving individual rationality. Aggregate queries are useful in uncovering common behaviors of moving objects, and their applications can be found in various fields such as transportation network design, tourism management, battlefield configuration, traffic control systems, and logistics management. However, traditional methods of spatiotemporal aggregation can be computationally intensive, as they require significant computational resources to obtain meaningful results using conventional database operations [2].

Our aim is to process the moving object query by reducing the communication overhead without compromising accuracy. The paper [5] introduces a method to identify vital locations using semantic constraints. Here, the query is executed over those selected areas, called semantic regions. To achieve this, the authors distinguish the set of aggregate queries applied to moving objects into two classes Static Spatio temporal Aggregate Queries (SAQ) and Continuous Spatio temporal Aggregate Queries (CAQ). The former category is analogous to traditional query, which executes the query and gets the result, whereas, in the latter, the evaluation of queries is continuous until a spatio-temporal limit is reached. The scope of CAQ is limited by two parameters such as time period and distance.

In a continuously moving environment, the optimization of query execution is crucial. This is especially important when a large number of queries are targeted at the same spatiotemporal domain. It is observed that moving queries can be generated from any point in a travel network. Many of the complex optimization techniques published suggest works on sub-expression levels for achieving better results [6]–[9]. These works exploit the moving micro-clustering method for managing objects and queries. The proposed algorithm generates separate clusters of moving queries according to their spatial and temporal interests. Also, simple techniques are adopted to group aggregate queries applied on similar spatio-temporal regions.

To implement the proposed models, two paradigms of data mining are utilized. They are semantic load shedding and shared cluster-based execution. The concept of load shedding has been studied in different contexts, such as networking, data stream management etc. [10]–[12]. It is the relinquishment of a fragment of raw data while processing the queries. There are algorithms to determine what point of data to load shedding at each point to reduce the degree of inaccuracy in the query output. For the purpose of this framework, the concept of semantic load shedding is used along with shared cluster-based execution. A number of load-shedding procedures are employed in data stream management systems with minimum loss in result accuracy[13]. According to various literature [14], [15], state-of-the-art load-shedding procedures are classified into random load shedding and semantic load shedding. In SCUBA, approximation of different object locations is made by semantic load shedding by avoiding less important data. It keeps track of the relative positions of objects in an area and is stored as a cluster nucleus. Based on the accuracy requirements, SCUBA adopts three ways of load shedding: full, partial, and no load shedding.

Shared cluster-based execution is the concurrent execution of continuous spatio-temporal queries. This concept has been adopted in many query execution models developed for diverse environments [16]–[18]. Similar queries are grouped together into the same data structure to achieve scalability. In the proposed model, query optimization is achieved by combining queries of similar spatio-temporal and semantic domain environments. While many ongoing research efforts focus on extracting knowledge from mobility data, integrating query processing with semantic features is a relatively underexplored area. To address this research gap, we propose an effective query-processing approach for spatio-temporal data using the SemTraClus model [5] as a baseline for our work. To the best of our knowledge, no existing work has incorporated semantic load shedding for processing moving object queries. We conducted experiments using both real and simulated datasets to achieve the following objectives.

- Define different classes of spatio-temporal aggregate queries and provide data representations for each.
- Proposes data structure for the representation of object representation in different clusters.
- Treat moving objects and queries as a single unit by implementing a shared cluster-based execution.
- Implement Semantic-based load shedding for leveraging computational overhead.
- Implementation of cluster-based algorithms to answer static and continuous spatio-temporal aggregate queries in an efficient manner.
- Evaluate the model using real-time data set to demonstrate the algorithm's efficiency.

The rest of the paper is organized as follows: In section 2, we narrate some of the relevant works in the related area; in section 3, basic concepts are briefed. Section 4 explains the methodology of prosed work; section 5 gives a detailed evaluation of various scenarios. In section 6, we conclude the topic by providing certain future directions.

2. Related Works

The relevant works in continuous clustering of moving entities and querying over spatio-temporal data are examined here—clustering groups similar elements together. A set of objects that move near in space for a long time period can be defined as moving clusters. Continuous clustering or incremental clustering is the clustering of elements in an interval of time or space. Continuous clustering, also known as incremental clustering, involves clustering elements in a specific interval of time or space. Micro clustering is a density-based algorithm used to obtain groups of strongly correlated classes of objects. As a concrete idea [19], this method has been exploited in different data management techniques like stream clustering. It is a temporal extension of cluster features. Moving micro clusters denotes groups of objects that are close not only to the current time but to the future spatial and temporal scope. Objects in these clusters may split or merge during the course.

Moving object clustering needs exceptional approaches due to its kinetic characteristic. One challenge is the difficulty in organising two-dimensional geographical information along with the temporal domain. Incremental clustering is a well-known solution in this regard, according to which locations are clustered only upon the updation from different moving objects. This avoids the re-clustering of the entire space-time components at every time unit. Incremental clustering desists the necessity of storing all location updates that, in succession, reduces storage requirements. Several constraints arise at this stage, particularly with the cluster's centroid, speed of movement, and termination condition. Literatures have suggested various ideas to decipher these curbs.

In a study, Jidong Chen et al. [19] argue that unlike the methodologies implemented over a static environment, additional measures are to be taken to unveil hidden patterns in the mobility data. This method, named moving micro clusters, is an extension of micro clustering. Here objects so close together are clustered, and the group of objects that are currently not together but likely to move together in the future are also considered. The object's profile is defined by time, position and velocity. The K-Means clustering is adopted here as objects with similar profile form clusters. It has a predefined velocity; if some variations above a threshold are defined, the current profile needs to be updated. One of the additional features of this method is the management of split events for the threshold values of a bounding rectangle.

The framework proposed by R.V Nehme et al. [4], called SCUBA, suggests the idea of grouping moving objects and queries in a single unit according to its spatial and temporal features. SCUBA adopts low-cost moving micro-clusters for the evaluation of spatio-temporal queries. The cluster memberships of each object are updated incrementally for every time unit. As the speed changes during the motion, the membership of individual objects in the cluster may vary. To retain the performance while processing multiple objects, authors have proposed a semantic load-shedding mechanism where the insignificant points are neglected. The innovative concept in this model is the arrangement of various object movements and queries in a single cluster. This method does not account for the splitting of moving objects in different clusters.

The same authors have put forward another proposal called ClusterSheddy [20], a semantic load-shedding-based continuous data processing approach. They have considered the application scenario of fleet monitoring and assume that vehicles move in convoys. In one approach, if the whereabouts of vehicles (e.g. vehicles with perishables of high value) is known, a record of such object will keep, and that of low

priority vehicle will be dropped at the time of load shedding. If such data is unavailable, vehicles exhibiting similar movement patterns will be accounted for, and their features will be constituted to form clusters.

Another method [19] proposes the clustering of moving objects in spatial networks. This framework, called CMON, performs periodical clustering of moving objects with different criteria. In order to preserve the clusters, it proposes a structure called cluster block throughout the session. The CMON method productively supervises the split and merge of cluster blocks. Intermediate destinations are kept in the travel path to manage the termination point. When the objects in the cluster block reach the destination, it departs from the cluster block. In order to reduce the cost of splitting based on the direction of movement and speed, the split scheme is managed. When neighbouring cluster blocks are moving together with a minimum distance threshold, they merge together, and that reduces cluster maintenance costs. The CMON framework does not manage queries on moving objects.

Jensen, C. S et al. [3] propose a strategy that is capable of clustering moving objects in an incremental fashion. The key properties of moving object clusters are set and updated incrementally by maintaining a special data structure called the clustering feature. Compared to the existing approaches, this method automatically detects cluster split events that eliminate the need to maintain bounding boxes of clusters with large amounts of associated violation events. As indicated in the introduction, micro clustering is a suitable data mining strategy for moving objects. It indicates that a group of objects so close to each other probably belongs to one cluster.

Li et al. [21] propose an incremental clustering framework called TCMM for a continuously updating system, assuming new locations will not affect clusters distant from the incoming data. It performs the operation in two phases. Initially, the micro clustering phase generates representative trajectory line segments by differentiating an incoming trajectory from the existing clusters. Similar micro-clusters are merged together in a periodical fashion. This helps to avoid the maintenance cost of unnecessary micro-clusters. Presented by Steven Young et al. [22], a fast and stable incremental clustering algorithm enforces a minimum memory requirement, utilizing the Winner Take All paradigm. This computational model is typically applied in neural networks for competitive learning. A supplementary statistical measure is taken to stabilize the parameters for the clustering process. Based on the parameters used for queries and outputs generated, queries are classified into the following categories [23].

2.1. Range Queries

These are basic operations on the trajectory database [23] that retrieve records within a specific range. Range queries on trajectories work on historical spatio-temporal data and count the number of distinct trajectories intersecting the query area [37]. For example, ‘get the details of containers passed through specific region’ or ‘get the list of cars crossed the location in a given period’.

2.2. Continuous Queries

These are issued once and evaluated incrementally until a termination criterion is reached. The termination is a spatial or temporal threshold or a combination of both. The outcome of incremental execution of queries is the same as the result of traditional user queries after each database update. But in the former case, the computational expense is substantially reduced [25].

2.3. K-Nearest-Neighbour Queries

Given a set of object locations L and a query Q, a K-nearest-neighbour query returns k closest locations $l1, l2, l3, l4 \ldots\ldots lk$ to Q [26]. For example, a query was issued by moving police to get the list of ambulances near it. Different alternatives to this have later been introduced. One continuous KNN query is a variant that identifies k- interested points along a trajectory [27]. For example, get the list of ambulances near its location from a source point to the destination point. Two, reverse KNN query describes the finding of the data objects with the query object in the set of their k-nearest neighbours. This is used to point out the influence of a query over a data set[28].

2.4. Aggregate Queries

Applied on a set of tuples and provide a single value that describes the nature of the selected group. The aggregation function we considered is SUM, MIN, MAX, AVG and COUNT. These fundamental queries are very useful and find their applications in understanding the general behaviour of a data set. Our model focuses on the effective evaluation of spatio-temporal aggregate queries. Considering the final inference based on the two abstractions, the clusters and aggregate queries provide the general behaviour of a set of data.

The discussion of constraint databases and constrained queries is relevant here as well. These ideas are not new; recently, these concepts have been studied in the context of new application domains such as moving object data management. A constraint database is an extension of the relational model representing a potentially infinite set of continuous data through finite combinations of polynomial equality and inequality constraints [29]. Different classifications are possible while discussing queries in the mobile environment, elaborated in the coming section.

3. Basic Concepts

The term "semantics" typically pertains to linguistic analysis and involves the study of the meaning and interpretation of words. However, in this context, it is used to

refer to the contextual information extracted from the trajectories of moving objects, which are referred to as "semantic properties." According to various applications, these features help us to arrive at many inferences about the moving entity and moving path. Here the authors refer to the moving entities as moving objects and moving object queries. Many works have been published for the extraction of semantic location extractions [5], [30]–[32], which all rely on application-dependent parameters.

The proposed work exploits semantic features of moving clusters and moving queries. Fundamental concepts on moving objects, trajectories, semantic properties etc., explained in SemTraClus are applicable in this context as well. The aim of our proposed work is to manage moving object queries efficiently. Concepts such as continuous clustering and semantic load shedding are discussed here.

3.1. Continuous Clustering of Moving Entities

Continuous clustering or incremental clustering is the grouping of moving objects with respect to the geographic distance over a constrained path. We consider moving objects and queries over a constrained path in the proposed model as moving entities.

A cluster at time slot tw is represented as $(\text{Cid}, \text{Clat}, \text{Clon}, \text{Ct}, \text{OSpeed}, \text{Ceps}, \text{Coids}, \text{Cqids})$, where Cid uniquely distinguishes instance of a cluster, Clat is the cluster latitude, Clon is the cluster longitude, Ct is the time at which cluster resides in the space, OSpeed is the speed of each object, Ceps is the radius of the cluster, Coids is the set of objects in the cluster, and Cqids is the set of queries attributed to the specific cluster.

Density-based incremental clustering in specified locations with respect to semantic properties such as speed and direction is performed[33]. Additionally, the semantic features computed from the traces are maintained in semantic structures such as *Obj_semantics* and *Clus_semantics*.

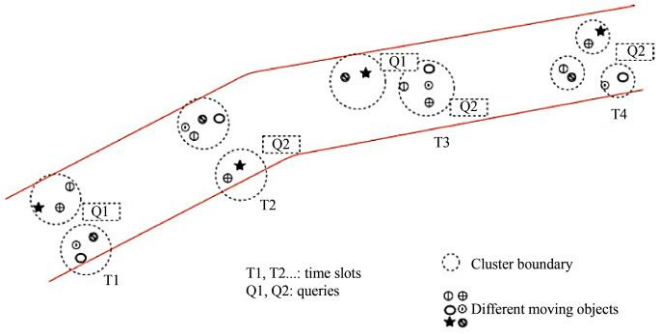

Fig. 1 Scheme of periodic clustering of moving objects and queries

The Spatio-Temporal Aggregate query for a given space-time constraint can be represented $\text{SQ}(\text{id}, \text{x}, \text{y}, \text{t}, \text{Attr}, \text{qtype})$. Given a tuple of attribute values $\text{Attr} = (\text{attr1 attr2} \ldots\ldots \text{attrd})$ where $\text{attri} \in \text{dom}(\text{Attr})$, the domain value specifies values for a select and conditional clause in the query, and it is the unique identification value for each query. Here (x, y) is the spatial component, and t is the temporal component, where $t_{i-1} < t_i < t_{i+1}$. The triplet (x, y, t) is termed a query feature. The parameter Attr specifies values for select and conditional clauses, and qtype specifies which category the query belongs to.

Figure 1 shows a model for the continuous clustering of moving entities. The proposed approach employs the SemTraClus method to identify cluster locations for clustering in different semantic locations. This allows for semantic load shedding using only the identified points of interest (POIs) for further processing rather than the entire records. Additionally, the proposal incorporates a shared cluster-based execution in which objects and queries are grouped together in each time slot. These two concepts are illustrated in the accompanying figure.

3.2. Query Types

As mentioned, the proposed work focuses on spatio-temporal aggregate queries. Two classes of spatio-temporal aggregate queries applicable to the moving objects are identified: Static spatio temporal Aggregate Queries (SAQ) and Continuous spatio temporal Aggregate Queries (CAQ)[34]. Static aggregate queries are instantaneous queries, while continuous aggregate queries require regular updating of the responses.

There are two expiration parameters applicable to CAQ, time period (Δt) and distance Δd. There are two expiration parameters applicable to CAQs: time period (Δt) and distance (Δd). Two different scenarios are illustrated in Figures 2 and 3. A query from a moving cab, 'How long from here to nearby hospital', is an example of a static spatio-temporal aggregate query.

Fig. 2 Static spatio temporal aggregate query scenario

Fig. 3 Continuous spatio temporal aggregate query scenario

Though the location of querying is important to retrieve the answer, the scope of the query ends after responding. While the query 'Update every 30 minutes number of emergency vehicles near to me' from a police cab clearly depicts the continuous spatio-temporal aggregate query. The execution proceeds until the expiration value is reached. Both categories of aggregate queries are executed in either an inter-cluster or intra-cluster fashion. In the former case, the semantic features of the individual moving objects are significant. In contrast, in the latter case, the semantic features of the cluster as a whole are the driving factor.

Special data structures are devised to store the cluster profiles, which monitor different parameters in various time slots. Queries are applied to this data structure. The query is registered to the pool of queries as soon as it is generated. This section groups queries in accordance with the intersection of spatial and temporal domains. The generation of static spatio-temporal aggregate query instances is pictured in Figure 4.

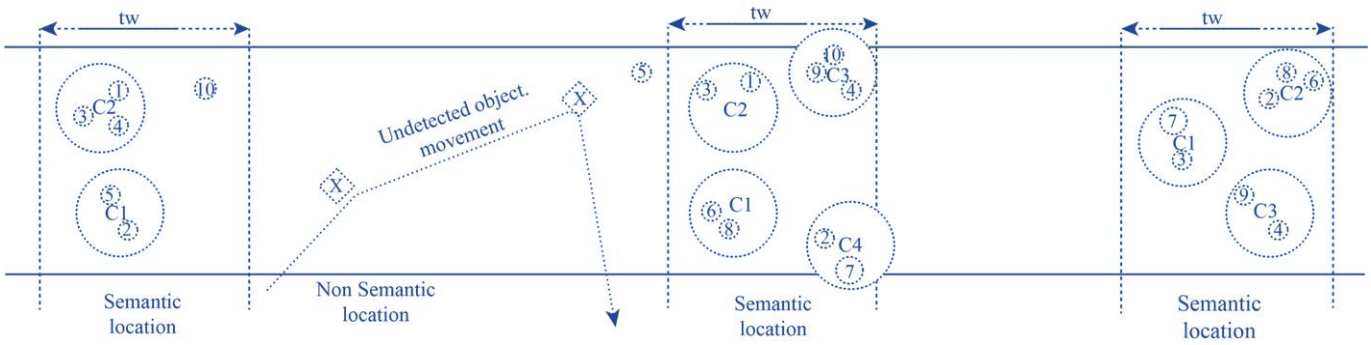

Fig. 4 Generation of static spatio-temporal aggregate queries

Some of the possible static spatio-temporal aggregate query instances

- Obtain the count of objects travelled between the given locations and during the given time

```
select count (*) from cluster_ds as c
where c.tim between t1 and t2
```

- Number of objects proceeding to loc1 travelling with the speed of speed

```
select count (*) from cluster_ds as c
where c.location = loc1 and
c.speed <= speed
```

Some of the possible Continuous Spatio Temporal Aggregate Query Instances

- Obtain the count of objects travelled between the locations until time1

```
select count (*)from cluster_ds as c
where c.location between loc1 and loc2
until t1;
```

- Average speed of objects traveling together between loc1 and loc2 for the time t.

```
select count(Objid), slot,
clusterIndex from clustered_table group by
slot, clusterIndex;
```

- Count of objects moving to loc1 with a given speed

```
select count(*) from cluster_ds as c
where c.location between loc1 and loc2
and c.speed <= speed until c.cur_time+t
```

- Number of Taxi cabs available in 200 meters with distance in next ten minutes between loc1 and loc2 travelling with the speed of speed.

```
select count (*), dist from cluster_ds
as c where c.location between loc1 AND
loc2 and c.speed <= speed unitl
c.curr_time + t
```

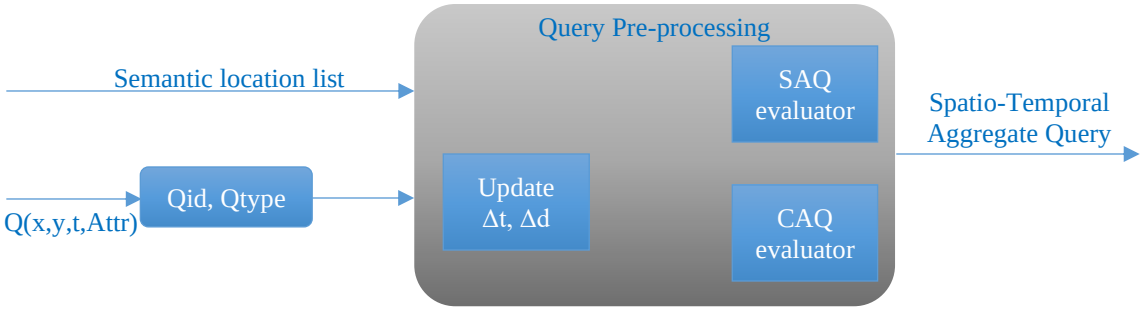

Fig. 5 Phases of spatio temporal aggregate query processing

4. Methodology

The queries are executed in a series of two steps. Initially, the query processor accesses query features and identifies the geographical and temporal sanctity of the query.

The qtype distinguishes it as inter-cluster or intra-cluster; accordingly, Δt expires, or Δd is calculated (as shown in Figure 5). Inter-cluster queries are queries that can be answered within the purview of clusters. While in the case of intra-cluster queries, more than one cluster is involved. Another input to the system is the list of semantic locations during these points till the expiration values results are returned in the case of a Continuous Spatio Temporal Aggregate Query.

4.1. Clustering Window

To process moving entities, we use a simple clustering window for a time slot *tw*. When a new moving entity is registered at *SemLoc*, the semantic attributes of the cluster at that point are re-estimated by taking into account the new member, such as the average speed of objects in different clusters, the number of objects moving together, and the maximum speed of an object between specific locations. A re-clustering is executed for all objects present at that instant. At this point, the following possibilities are applicable with respect to the time window.

- New clusters are generated: - If a new object appears or an existing object moves away from the cluster.
- Existing clusters are merged: - When inter-cluster distance reduces due to the change of location.
- Change of the cluster radius: - Cluster radius can be increased or decreased according to the geographical area within the limit of the maximum threshold.

Consider the illustration given in Figure 6. Here, the movement of ten objects is considered, and clustering is applied in the specific geographical area (clustering window, τ) with time slot *tw*. Along the path, each object occupies a different cluster, which means the object of cluster membership varies during the course of the journey. In the first slot, six objects are spread among three clusters. Obj_10 is the sole element in a cluster. Obj_5 leaves the path before the next clustering location. More objects are added at the second time slot, and the cluster count increases.

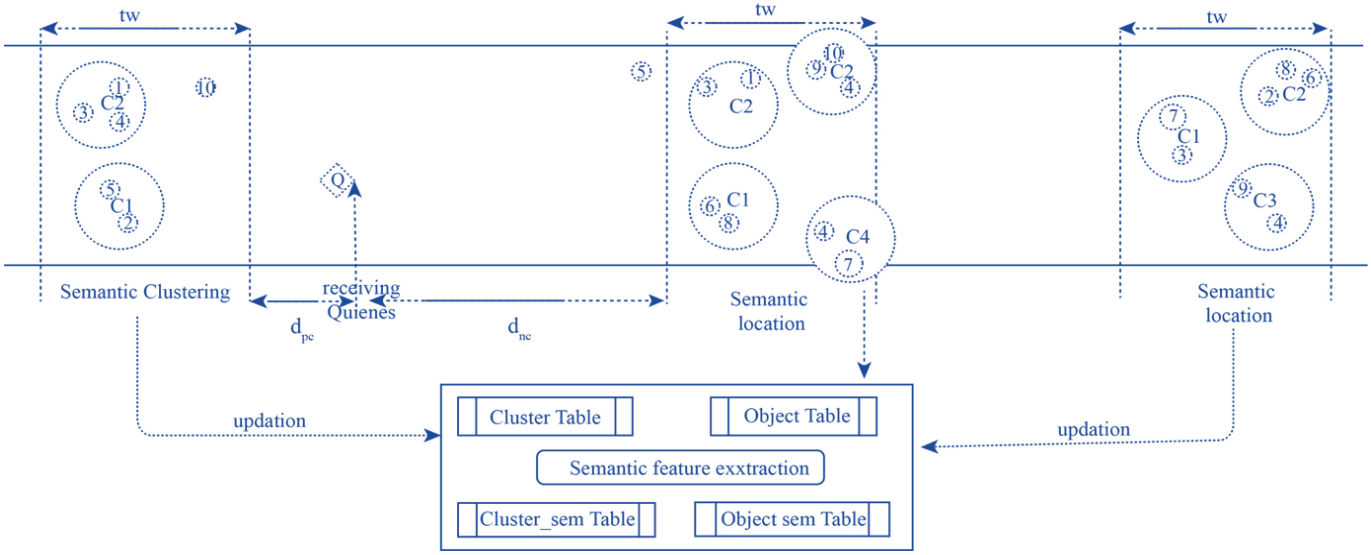

Fig. 6 Periodic clustering of moving objects and queries

Q is a query generated between two cluster slots. Then the system calculates the distance between the current position of the query and the immediate cluster window (this could be an already generated cluster (previous cluster pc) or a forthcoming cluster (next cluster nc) slot). The answer will be retrieved from a short distant point and based on the proximity thresholds. If the last updated slot and next updated slots are above the threshold values, the system alerts for an immediate clustering request.

4.2. Semantic Load Shedding and Model for Representing Continuous Clusters

In order to reduce the processing cost of queries and manage latency, many solutions, including lazy updates, selective reduction of tuples etc., are employed. This technique, load shedding, has become a necessary requirement where approximate answering of queries is required, like dynamic processing of moving object queries. At this point, we are providing certain alternative forms of overhead deduction with density-based clustering. Object properties in semantic locations with a timing window are considered for processing. Semantic features of the location are also recorded. In order to manage object movement among different clusters in semantic load shedding, the entire cluster is considered a moving object.

To evaluate continuous spatio-temporal aggregate queries, periodical clustering is executed in every semantic location in a cluster for a time slot with duration Δtw. According to the mobility behaviours, different numbers of clusters are generated in each cluster slot. Here the cluster slot is an identified semantic location SemLoc(x,y). The cluster membership of each object changes from SemLoci to SemLoci+1. The transition is accordingly with its semantic attributes such as velocity, the structure of the constrained path, direction of movement etc. Though objects perform constrained movement, their mobility pattern can change within limits stipulated by the travel network. These changing mobility itself will give the travel patterns of the object. In order to represent changing mobility of objects, we represent them in a tabular form as modelled in a related study published in [35].

This representation is illustrated by considering the movement of ten objects along a path. O = {o1, o2, o3, o4, o5, o6, o7, o8, o9, o10}, these are clustered over four consecutive semantic locations. After clustering, the objects are scattered in different clusters. In one such instance let,

$$SemLoc_1 = \{(o_1,o_2,o_3), (o_4, o_5, o_6), (o_7, o_8, o_9)\},$$

$$SemLoc_2 = \{(o_1, o_3, o_5), (o_4, o_6), (o_7, o_8), (o_2, o_9)\},$$

$$SemLoc_3 = \{(o_2, o_5), (o_1, o_3, o_6), (o_7, o_8), (o_4), (o_{10})\},$$

$$SemLoc_4 = \{(o_2, o_5, o_{10}), (o_1, o_6), (o_3, o_7, o_8), (o_4)\}.$$

Table 1 displays the semantic locations and their corresponding clusters, with each row representing an object and each column representing a semantic location. The variable 'i' represents the cluster to which the object belongs in the corresponding semantic location. The final column shows the feature (F) of the object, which is its cluster in the entire movement session. This representation effectively tracks the movement of objects across different clusters and can be used for analysing spatiotemporal aggregate queries.

The frequency and places of clustering are predetermined using the SemTraClus [5] method, identifying potentially interesting regions, such as significant traffic junctions, tourist spots, and diversions, in the case of a road network. The arrival of new objects and the departure of existing ones are noted at this stage.

4.3. Shared Cluster-Based Execution

In a continuously moving environment, optimization of query execution is crucial, especially when more queries are targeted on the same spatio-temporal domain. It is observed that moving queries can be generated from any point in a travel network.

Table 1. Cluster membership of different objects in various semantic locations

Objects	$SemLoc_1$	$SemLoc_2$	$SemLoc_3$	$SemLoc_4$	F
o_1	1	1	2	2	1
o_2	1	4	1	1	1
o_3	1	1	2	3	1
o_4	2	2	4	4	2,4
o_5	2	1	1	1	1
o_6	2	2	2	2	2
o_7	3	3	3	3	3
o_8	3	3	3	3	3
o_9	3	4	_	_	0
o_{10}	_	_	5	1	0

Many of the complex optimization techniques published suggest works on sub-expression levels for achieving better results [6]. SCUBA exploits the moving micro-clustering method for managing objects and queries. The proposed algorithm generates separate clusters of moving queries according to their spatial and temporal interest. Also, simple techniques are adopted to group aggregate queries applied on similar spatio-temporal regions.

Consider Figure 7, where Q1, Q2 and Q3 are different aggregate queries generated from various sources that share the same temporal domain. As a result, for the time period from t2 to t7, all these queries can be grouped together for processing.

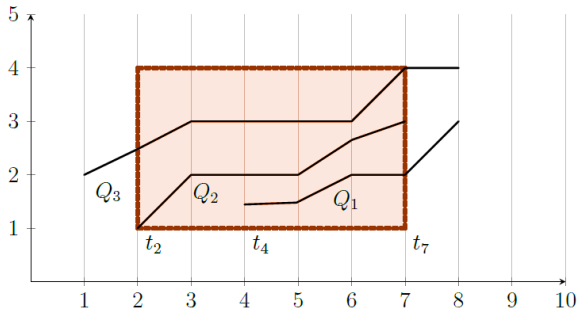

Fig. 7 State diagram of shared cluster-based execution of spatio-temporal aggregate queries

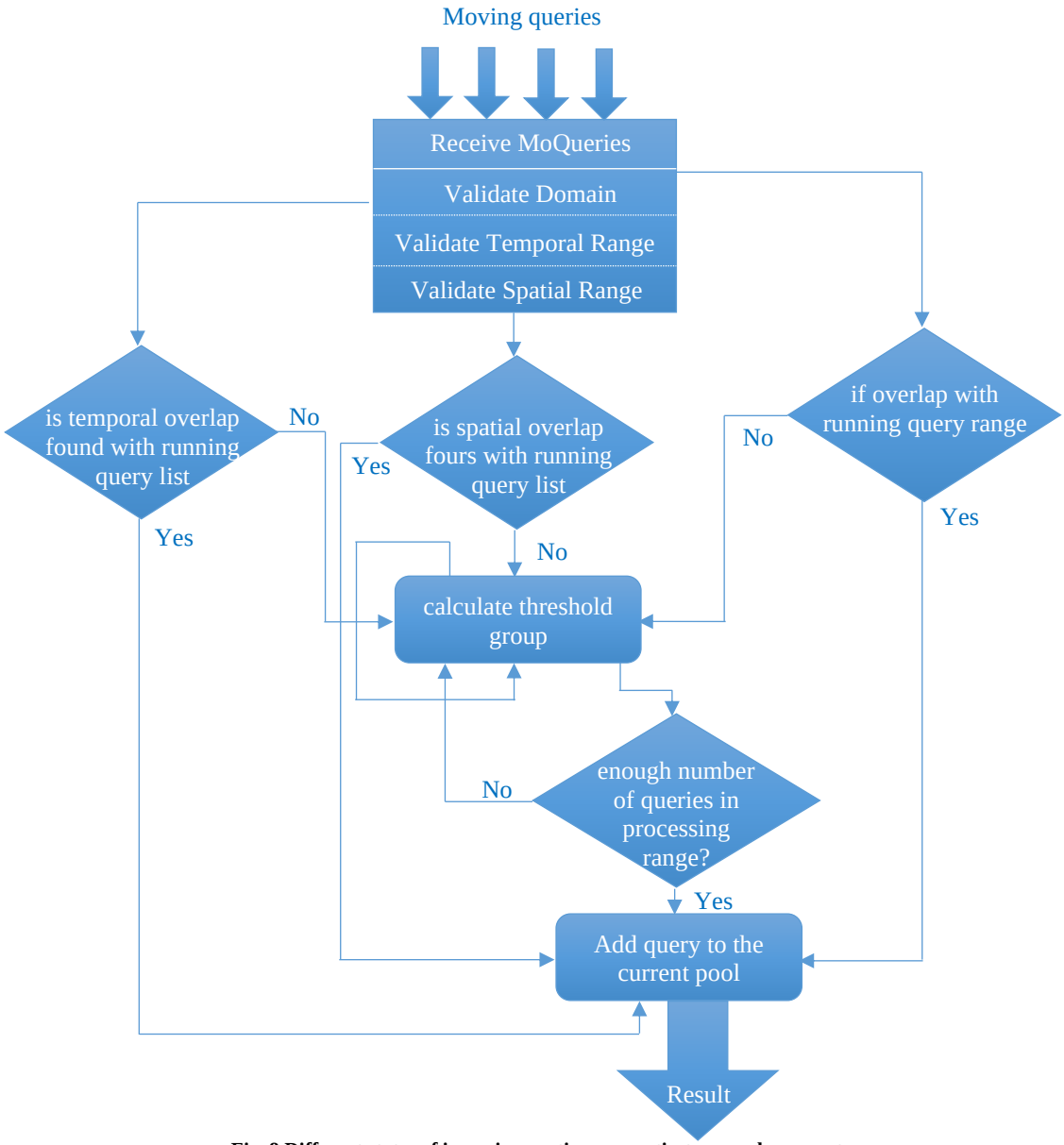

Fig. 8 Different states of incoming continuous spatio-temporal aggregate query

This is known as shared cluster-based execution, which involves concurrently processing continuous spatiotemporal queries. This approach has been adopted in many query execution models developed for different environments [16], [17]. By grouping similar queries together, scalability can be achieved. In the proposed model, query optimization is achieved by combining queries with similar spatiotemporal and semantic domains. If queries with attribute values from the same domain are being processed, they are grouped together for shared segments.

Figure 8 illustrates the different states of incoming continuous spatiotemporal aggregate queries. Queries can be received at any time from various sources, and their evaluation is abstracted as a spatial join between the objects table and the query table. If the response interval is above the threshold interval, the query execution is postponed until the next immediate clustering location. Otherwise, the query is executed based on previous values, representing a compromise between performance and response time.

Once a query is registered, three types of validation occur in the system: domain validation, spatial range validation, and temporal range validation. In the domain validation phase, the system checks if the domain of the incoming query is suitable for grouping with any of the running instances, considering simple query types. In the spatial range validation, the system checks the spatial scope of the incoming query by validating the spatial expiration distance. The system checks the temporal distance expiration value in the temporal range validation. Finally, different queries are pooled together and executed.

5. Evaluations

5.1. Environment Setup

The BerlinMOD[36] synthetic traffic data set and Microsoft TDrive real data sets were used in the experiment. 21 different moving objects along a constrained transportation network were considered. The performance of both SAQ and CAQ was evaluated using the DBSCAN method tuned for incremental clustering. The two key parameters, Eps and MinPt, significantly influence the inclusion of points in clusters, so choosing optimal values is crucial. A constant value for Eps is not feasible as the objects continuously change their locations. A low value of the MinPts causes the inclusion of noise points in the cluster. For selecting Eps and MinPts, this work followed the common notion specified in the DBSCAN method. The heuristic approach in [30] was used to determine MinPts, which proposes that it should be equal to ln(n), where n is the number of points selected for clustering. To choose the cluster radius Eps, the sorted distance of each point to its k-nearest neighbors was plotted, where k was considered to be MinPts. The Eps value was selected from the knee point of the graph. The experiment was conducted using randomly generated environments of static and dynamic spatiotemporal aggregate queries.

Object traces are stored in MySQL database version 8.0.14. Aggregate functions in RDBMS are used for providing summary data. MySQL defines a number of aggregate functions suitable in different data mining environments such as MIN (), MAX (), SUM (), AVG (), COUNT (), STD (), GROUP_CONCAT (), VARIANCE () etc. In fact, with respect to our application domain, we consider the first five standard aggregate functions listed here for our application. For the implementation purpose, we use spatial extensions of MySQL. It provides an SQL environment that has been extended with a set of geometry types. This package provides data types such as GEOMETRY, POINT, LINESTRING and POLY GON and methods such as ST_Contains(), ST_Distance() etc.

5.2. Experimentation Statistics

Table 2 details the evaluation statistics of two different scenarios, first, with varying numbers of objects in the transportation network (Table 2a). Here the number of records, the maximum number of clusters generated and the maximum Eps values are listed. A large value in the maximum number of clusters depicts a vast geography of the transportation network. The maximum number of semantic slots is limited to 250 in this context. Consider the instance where MaxObjCount = 15, SemLocCount = 250 maximum number of clusters formed throughout the journey is 5, found in 32 different clustering slots. The maximum number of objects that appeared in a cluster is 4. On close monitoring, it is found that three objects travelled throughout all the clustering locations. Single objects in a cluster are considered an outlier and omitted from the further calculation as they do not contribute to the result.

Table 2. Data set description and cluster formation statistics

(a) Cluster slots = 250

No. Objects	No. Records	Max. No. Clusters	Eps
5	3895	2	1.52
10	13411	3	4.512
15	22026	5	6.253
20	39157	7	7.953
25	45076	9	6.4

(b) Object count = 50

No. Time Slots	No. Records	Exec. Time (s)	Max. No. Clusters
50	3673	0.56	2
100	7273	0.47	3
200	14545	3.5	3
500	35321	17.88	4
1673	49889	33.52	3

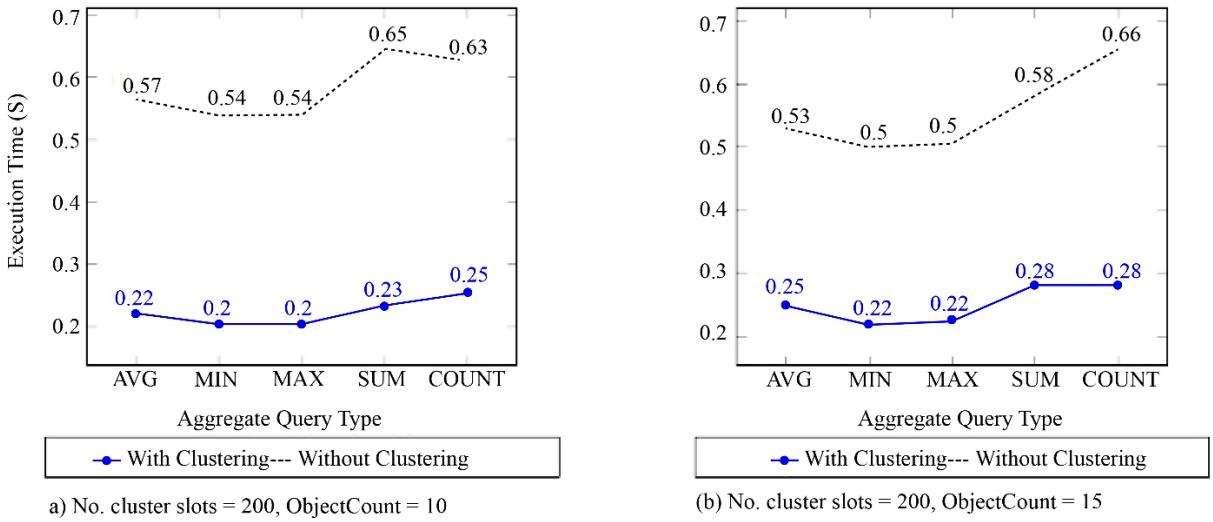

Fig. 9 Comparative plot on the execution time for different aggregate queries

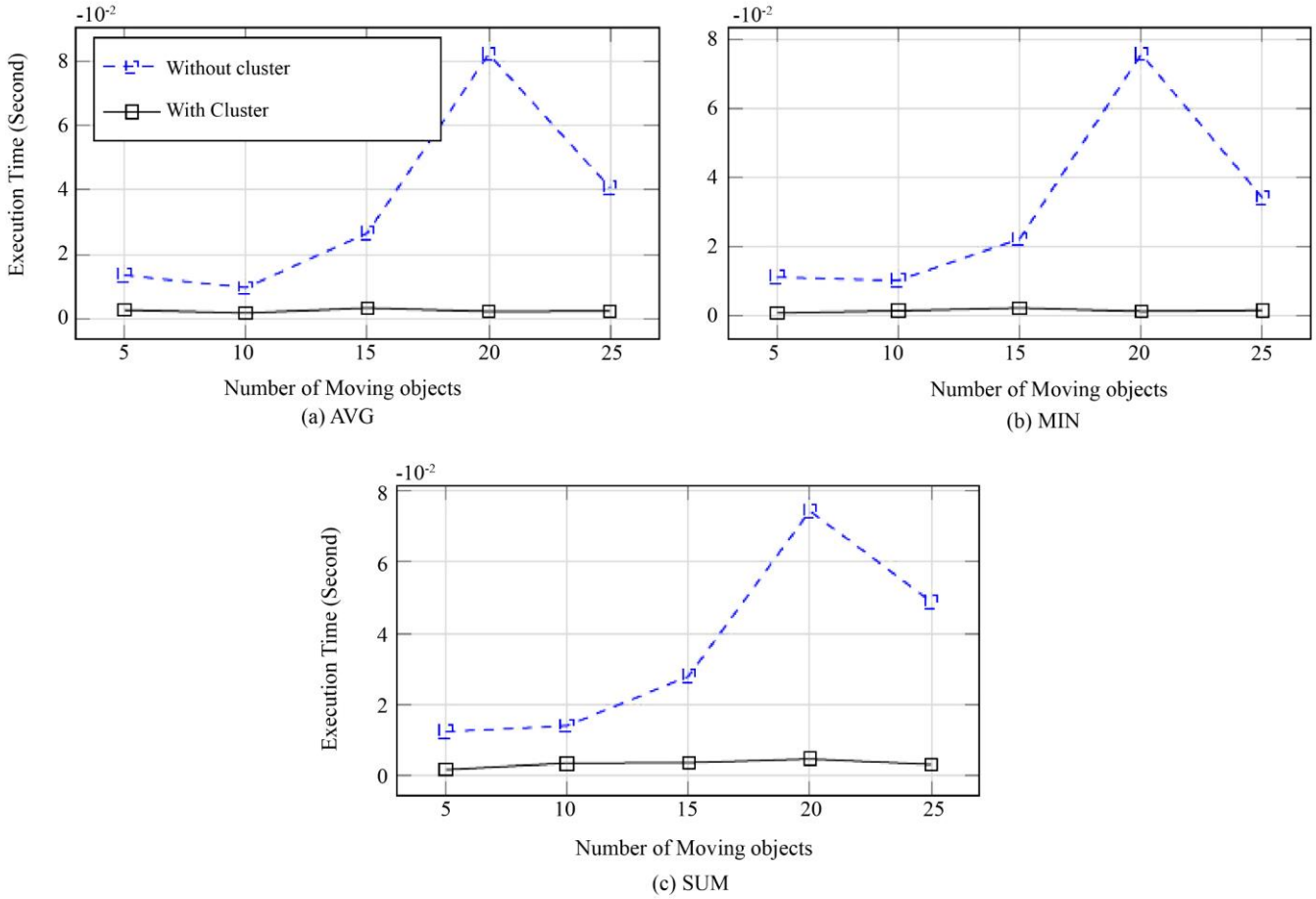

Fig. 10 Execution time of SAQ with different Object count, POI = 200

In the second scenario (Table 2b), the details of cluster formation and execution time are listed for ten objects and varying numbers of semantic slots. The clusters consist of varying numbers of objects, and the membership of objects in each cluster changes as they move towards their destination. Single-object clusters account for no more than 10% of the total number of clustering slots. As the number of semantic locations for clustering increases, so does the execution time.

The number of clusters generated is dependent on the pattern of object movement. Fewer groups indicate that the objects move closer together and have less variation in speed. The number of clusters also depends on the geographic nature of the trajectory. During the ride, part of the moving object may depart from the path. The cluster structures formed at this stage are stored in a database to avoid heavy computation processes and to answer queries efficiently. Figure 9 shows the time required to execute various aggregate queries with varying object counts and cluster slots. In the figure, dashed lines indicate the traditional approach, i.e., the application of queries on entire data, while the solid line indicates our approach.

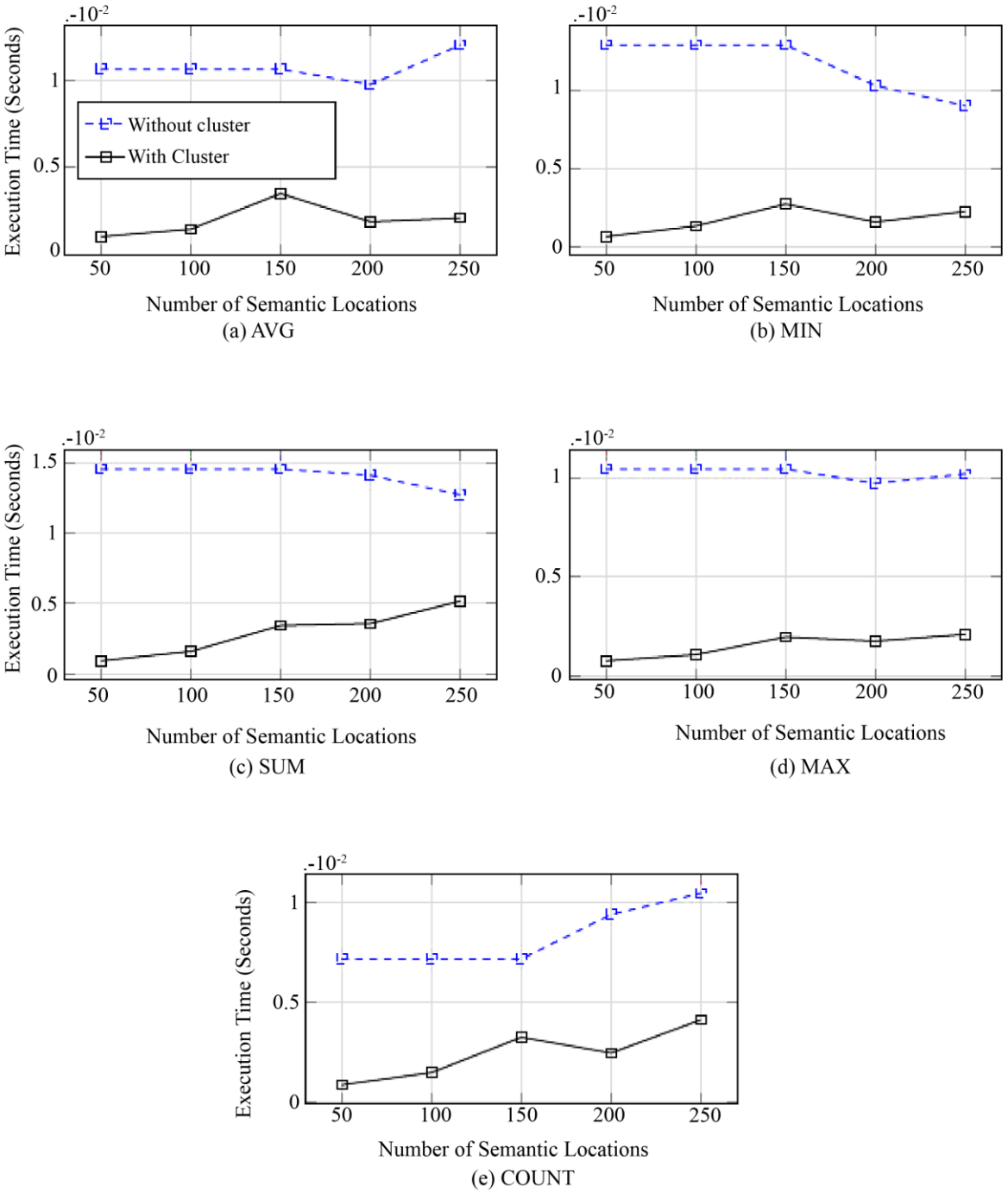

Fig. 11 Execution time for aggregate queries for various numbers of semantic locations

5.3. Varying Object Count

The performance of our approach is analysed with different numbers of objects monitored in the transport network. Figure 10 plots the execution time required for three aggregate queries in the proposed approach (solid lines) and traditional method (dashed lines). In order to incorporate the traditional method, the whole trajectory points were considered for executing the query. All queries consume comparatively the same amount of time for execution.

5.4. Varying Semantic Location Count

Another analysis to showcase the performance of the approach is based on the execution of different queries with different numbers of semantic locations. Figure 11 illustrates the execution of queries in varying clustering locations with a constant object count. Here the impact of varying the number of semantic locations in spatio-temporal aggregate queries is analyzed. It can be done in two ways, either by increasing the cluster frequency or increasing the trajectory length. For the inclusion of better cases, the authors simulated a mixed scenario. As mentioned above, the framework addresses inter-cluster and intra-cluster queries—increasing the number of clustering locations. The comparative plot of query execution time for SAQ and CAQ conveys the same meaning; hence, the experimentation of SAQ is only included here.

5.5. Execution Time CAQ

Figure 12 provides a comparative plot for a set of CAQs. It is clear from the plots that as we increase the width of the slot, the time of execution also increases. Accuracy also increases accordingly with tw up to a point. In this experiment, the configuration with slot width = 200 performs better as it is the maximum point of the steady increase in time, after which a spike in the execution time can be seen for all the query types. Here, the non-clustering approach indicates the traditional way of query evaluation, but no clustering is performed. So the model accounted for all the location points up to a location for analysing the incoming query. It takes cumulative time to process. While the clustering method shows steady progress in the execution time compared to the non-clustering method, it is cheap with less cost of accuracy. Cluster radius is a crucial parameter that determines the accuracy of aggregate query results. The queries fetch data based on the semantic properties of objects and locations. These are generated using the SemTraClus algorithm. The parameters are dynamically fetched from their semantic properties. At run time, the cluster parameters are computed based on the semantic properties of the road network and the number of objects. The data structure is constantly updated with the spatio-temporal values and other parameters of the moving objects and the queries. However, having a large number of clusters can negatively impact the accuracy of the object mobility pattern, which can also depend on the width of the transportation network.

5.6. Aggregate Query Comparison

Another performance evaluation is illustrated in Table 3. As explained in the previous section, two factors are involved in the execution, namely, the time required for clustering and the query execution time. In our method, the clustering is performed in predefined locations called slots. The method reduces the overhead of managing entire points of the object movement that is added to the database. Consider a case of objctCnt = 10 and POI = 200; the cluster time for an individual slot is .0031 seconds, query execution time for Max = .0017339 s total time is 0.0048339, without clustering method has query execution time itself is 0.0097216seconds, which itself is 50% more than with clustering method. Another measure to indicate the effectiveness is detecting the objects' ceased/joined rate; Table 4 shows that out of a maximum of ten objects that participated in the process, only four objects proceeded till the end of the journey. Six objects either joined after the journey's beginning or ceased movement before reaching the destination. Our study could spot only five of these objects which exhibited this nature. The information loss is measured by detecting the ceased /joining rate of objects. As we are applying clustering on vital semantic locations, only that less number of objects ceased /joined the travel network without being noticed.

5.7. Detection of Objects in Various Slots

Table 5 gives the statistics of object occurrence in different clusters. Consider the instance with cluster width 250; different attributes of the same object are considered 11 times (maximum case) and 2 (minimum case); most occurrences are 7 at this particular slot. As the width of the cluster window increases, more instances of the same objects are included in a slot. Hence the accuracy also increases. However, increasing the slot width beyond a point is not feasible, as it is considering the semantic nature of a location.

It will also reduce the benefit of the incremental clustering approach. If more number of dynamic queries are reported from a non-semantic location, and if the next semantic location is not so near, the algorithm will execute with short width. Even though clustering is applied to semantic points, we have to filter out the locations based on performance evaluation. From the table, it is clear that if the cluster interval is increased after a certain point, the time required for execution increases, which costs the performance an optimum value considered as the clustering period. Another evaluation is based on the number of queries pooled for the execution.

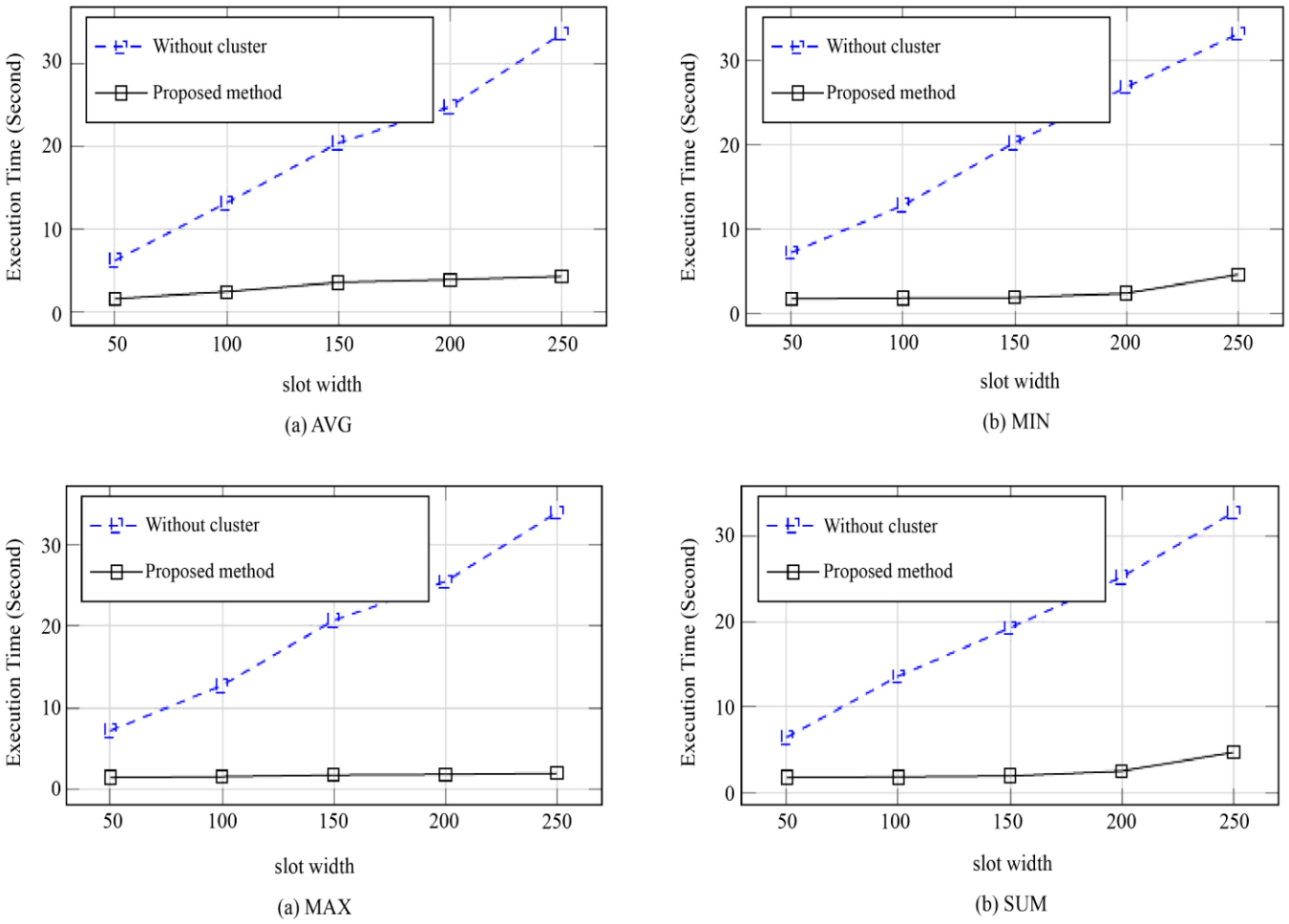

Fig. 12 Execution time for spatio-temporal continuous queries with varying slot width

Table 3. Execution time required for two types of queries objCnt = 10

Query Type	Clustering Slots	Execution Time (s) Witclusteringg			Execution Time (s) Withouclustering		Percentage of variance
		Query Exec.	Clustering	Total	Query Exec	Total	
MAX	200	0.0017339	0.0031	0.0048339	0.0097216	0.0097216	50.27670342
SUM	50	0.000949438	0.0063	0.007249438	0.0141021	0.0141021	48.59320598

Table 4. Detection rate of joining-attrition of objects

No. Objects	No. of objects detected (Ceased/ Joined)		Rate of variation
	With Clustering	Without Clustering	
10	5	6	0.166666667
15	11	13	0.153846154
20	15	15	0
25	21	22	0.04545454
30	23	26	0.115384615

Table 5. Number of objects repeated in slots for different width

Slot Width	No. of occurrence of objects in a slot		
	Min	Max	Mod
50	2	6	2
100	2	6	2
150	1	6	2
200	10	20	4
250	11	2	7

5.8. Shared Cluster-based Execution for CAQ

Queries aimed at similar spatial and temporal domains are grouped together after performing a series of validations. Figure 13 shows the number of queries received at a point and the actual number of queries executed after the clustering process. The difference between the two lines is the number of queries saved from execution due to concurrent execution. If more than one query on the same geographic location arises, such queries are pooled based on the domain. During execution, the authors omitted slots with single objects. Such instances are more while the slotting window is very low value. As the minimum number of objects in a cluster is set to ln (object_count), such slots do not add anything to the information. It is also noted that the execution time slightly varies in different slots according to the number of objects present at that particular instance.

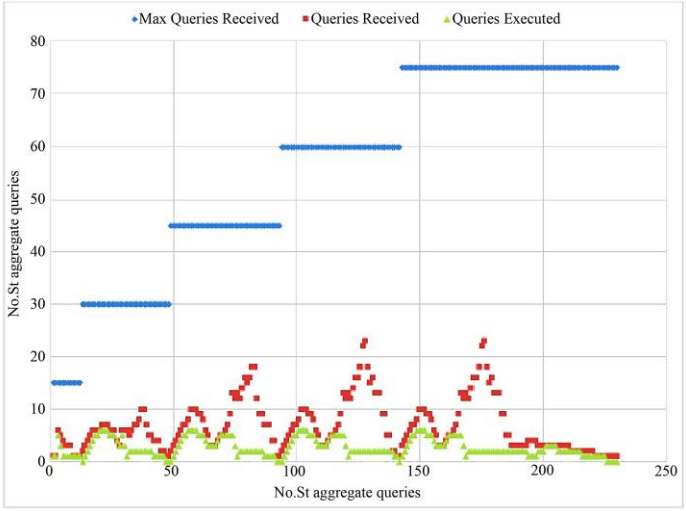

Fig. 5 Concurrent execution of queries in a shared environment

6. Conclusion

The clustering of moving object data is receiving more and more research attention as numerous applications generate vast amounts of spatio-temporal data from various location sensing devices. Query processing in a spatiotemporal environment has a different dimension from that of traditional processing. Fetching of summarized information about moving objects finds scope in a large number of applications like traffic summarization, for instance, the number of gasoline containers passing through a residential area in a specified time. The proposed algorithm is designed to handle spatio-temporal aggregate queries effectively. The algorithm relies on confirming that inferences derived from clustering and processing of aggregate queries yield the same insights. Utilizing an incremental density-based clustering process, the approach can efficiently answer spatio-temporal aggregate queries without needing to process the entire trajectory record. This approach accounts for moving objects' spatial, temporal, and semantic aspects. Evaluations of algorithms using various parameters are also detailed. Continuous queries are an effective way of extracting information from the mobility data set. Unlike traditional queries, it keeps on generating answers until a given criteria is met. Processing moving objects' data is a need of time due to the wide acceptance of context-sensing devices. In order to evaluate continuous dynamic queries, processing of entire GPS records is not feasible where time plays a crucial role. Here queries and objects are combined together with spatial and temporal abstraction. Density-based clustering is executed at semantic areas so that the aggregate values of points with potential interests can be easily ascertained. The authors also suggested pooling moving object queries working on the same spatio-temporal domain. Experiments show that incremental queries can be efficiently evaluated using the proposed methods.

References

[1] A. Nishad, and Sajimon Abraham, “Contact Tracing and Mobility Pattern Detection during Pandemics – A Trajectory Cluster Based Approach,” *International Journal of Pervasive Computing and Communications*, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[2] Yufei Tao, and Dimitris Papadiass, “Historical Spatio-Temporal Aggregation,” *ACM Transactions on Information Systems (TOIS)*, vol. 23, no. 1, pp. 61–102, 2005. [CrossRef] [Google Scholar] [Publisher Link]

[3] Christian S. Jensen, Dan Lin, and Beng Chin Ooi, “Continuous Clustering of Moving Objects,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 9, pp. 1161–1174, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[4] Rimma V. Nehme, and Elke A. Rundensteiner, "SCUBA: Scalable Cluster-Based Algorithm for Evaluating Continuous Spatio-temporal Queries on Moving Objects," in *Advances in Database Technology - EDBT 2006*, Y. Ioannidis, M. H. Scholl, J. W. Schmidt, F. Matthes, M. Hatzopoulos, K. Boehm, A. Kemper, T. Grust, and C. Boehm, Eds., in Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 1001–1019, 2006. [CrossRef] [Google Scholar] [Publisher Link]

[5] A. Nishad, and Sajimon Abraham, "SemTraClus: an algorithm for clustering and prioritizing semantic regions of spatio-temporal trajectories," *International Journal of Computers and Applications*, vol. 43, no. 8, pp. 841–850, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[6] Yousuke Watanabe, and Hiroyuki Kitagawa, "A Multiple Continuous Query Optimization Method Based on Query Execution Pattern Analysis," *Database Systems for Advanced Applications*, Y. Lee, J. Li, K.-Y. Whang, and D. Lee, Eds., in Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, pp. 443–456, 2004. [CrossRef] [Google Scholar] [Publisher Link]

[7] Medhane Darshan Vishwasrao, and Arun Kumar Sangaiah, "ESCAPE: Effective Scalable Clustering Approach for Parallel Execution of Continuous Position-Based Queries in Position Monitoring Applications," *IEEE Transactions on Sustainable Computing*, vol. 2, no. 2, pp. 49–61, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[8] Pingfu Chao et al., "Efficient Trajectory Contact Query Processing," *Database Systems for Advanced Applications*, C. S. Jensen, E.-P. Lim, D.-N. Yang, W.-C. Lee, V. S. Tseng, V. Kalogeraki, J.-W. Huang, and C.-Y. Shen, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 658–666, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[9] Joachim Gudmundsson, Patrick Laube, and Thomas Wolle, "Movement Patterns in Spatio-Temporal Data," *Encyclopedia of GIS*, vol. 726, p. 732, 2008. [Google Scholar] [Publisher Link]

[10] B. Babcock, M. Datar, and R. Motwani, "Load Shedding for Aggregation Queries Over Data Streams," *Proceedings 20th International Conference on Data Engineering*, pp. 350–361, 2004. [CrossRef] [Google Scholar] [Publisher Link]

[11] Zhang Longbo et al., "Semantic Load Shedding for Sliding Window Join-Aggregation Queries over Data Streams," *2007 International Conference on Convergence Information Technology*, pp. 2152-2155, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[12] Mario José Diván & María Laura Sánchez Reynoso, "A Load-Shedding Technique Based on the Measurement Project Definition," *Recent Trends in Intelligent Computing, Communication and Devices*, pp. 1027-1033, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[13] E. N. Tatbul, "*Load Shedding Techniques for Data Stream Management Systems*," Ph.D. Thesis, Providence, RI, USA. AAI3272068, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[14] J. Park and H. Cho, "Semantic Load Shedding for Prioritized Continuous Queries over Data Streams," *Computer and Information Sciences - ISCIS 2005*, pInar Yolum, T. Güngör, F. Gürgen, and C. Özturan, Eds., in Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, pp. 813–822, 2005. [CrossRef] [Google Scholar] [Publisher Link]

[15] Fethi Belghaouti et al., "Graph-Oriented Load-Shedding for Semantic Data Stream Processing," *2015 International Workshop on Computational Intelligence for Multimedia Understanding*, pp. 1-5, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[16] Xiaopeng Xiong et al., "Scalable Spatio-Temporal Continuous Query Processing for Location-Aware Services," *Proceedings 16th International Conference on Scientific and Statistical Database Management*, pp. 317–326, 2004. [CrossRef] [Google Scholar] [Publisher Link]

[17] X. Xiong, M. F. Mokbel, and W. G. Aref, "SEA-CNN: Scalable Processing of Continuous K-Nearest Neighbor Queries in Spatio-Temporal Databases," *21st International Conference on Data Engineering*, pp. 643-654, 2005. [CrossRef] [Google Scholar] [Publisher Link]

[18] Jianjun Chen et al., "Niagaracq: A Scalable Continuous Query System for Internet Databases," *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, in SIGMOD '00. New York, NY, USA: Association for Computing Machinery, pp. 379-390, 2000. [CrossRef] [Google Scholar] [Publisher Link]

[19] Jidong Chen et al., "Clustering Moving Objects in Spatial Networks," *Advances in Databases: Concepts, Systems and Applications*, R. Kotagiri, P. R. Krishna, M. Mohania, and E. Nantajeewarawat, Eds., in Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 611–623, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[20] Rimma V. Nehme, and Elke A. Rundensteiner, "ClusterSheddy: Load Shedding Using Moving Clusters over Spatio-temporal Data Streams," *Advances in Databases: Concepts, Systems and Applications*, R. Kotagiri, P. R. Krishna, M. Mohania, and E. Nantajeewarawat, Eds., in Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 637–651, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[21] Zhenhui Li et al., "Incremental Clustering for Trajectories," *International Conference on Database Systems for Advanced Applications*, Springer, pp. 32–46, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[22] Steven Young et al., "A Fast and Stable Incremental Clustering Algorithm," *2010 Seventh International Conference on Information Technology: New Generations*, pp. 204–209, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[23] Jianqiu Xu, Hua Lu, and Ralf Hartmut Güting, "Range Queries on Multi-Attribute Trajectories," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 6, pp. 1206–1211, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[24] M.S. Vinmathi, V. Sathiya, and M. Maheswari, "A Predictive-Reactive Procedure for Improving the Strength of Simultaneous Data Services," *SSRG International Journal of Computer Science and Engineering*, vol. 1, no. 10, pp. 1-5, 2014. [CrossRef] [Publisher Link]

[25] Douglas Terry et al., "Continuous Queries Over Append-Only Databases," *ACM SIGMOD Record*, vol. 21, no. 2, pp. 321–330, 1992. [CrossRef] [Google Scholar] [Publisher Link]

[26] Dimitris Papadias, "Nearest Neighbor Query," *Encyclopedia of Database Systems*, L. LIU and M. T. ÖZSU, Eds., Boston, MA: Springer US, p. 1890, 2009. [CrossRef] [Publisher Link]

[27] Mohammad R. Kolahdouzan, and Cyrus Shahabi, "Alternative Solutions for Continuous K Nearest Neighbor Queries in Spatial Network Databases," *GeoInformatica*, vol. 9, no. 4, pp. 321–341, 2005. [CrossRef] [Google Scholar] [Publisher Link]

[28] Elke Achtert et al., "Efficient Reverse k-Nearest Neighbor Estimation," *Informatik - Forschung und Entwicklung*, vol. 21, no. 3, pp. 179–195, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[29] Gabriel Kuper, Leonid Libkin, and Jan Paredaens, *Constraint Databases*, Springer Science & Business Media, 2013. [Google Scholar] [Publisher Link]

[30] Martin Ester et al., "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, 1996. [Google Scholar] [Publisher Link]

[31] Luis Otavio Alvares et al., "A Model for Enriching Trajectories with Semantic Geographical Information," *Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems*, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[32] Lucas Santos De Oliveira, Pedro O. S. Vaz-De-Melo, and Aline Carneiro Viana, "Assessing Large-Scale Power Relations among Locations from Mobility Data," *ACM Transactions on Knowledge Discovery from Data*, vol. 16, no. 2, pp. 1-31, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[33] Li Wan et al., "Density-Based Clustering of Data Streams at Multiple Resolutions," *ACM Transactions on Knowledge Discovery from Data*, vol. 3, no. 3, pp. 1-28, 2009. [CrossRef] [Google Scholar] [Publisher Link]

[34] A. Nishad, and Sajimon Abraham, "A Method for Continuous Clustering and Querying of Moving Objects," *Advanced Informatics for Computing Research*, A. K. Luhach, D. S. Jat, K. B. G. Hawari, X.-Z. Gao, and P. Lingras, Eds., in Communications in Computer and Information Science. Singapore: Springer, pp. 172–183, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[35] Aristides Gionis, Heikki Mannila, and Panayiotis Tsaparas, "Clustering Aggregation," *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[36] Christian Düntgen, Thomas Behr, and Ralf Hartmut Güting, "BerlinMOD: A Benchmark for Moving Object Databases," *The VLDB Journal*, vol. 18, no. 6, pp. 1335-1368, 2009. [CrossRef] [Google Scholar] [Publisher Link]

[37] Soheila Ghane, Lars Kulik, and Kotagiri Ramamoharao, "A Differentially Private Algorithm for Range Queries on Trajectories," *Knowledge and Information Systems*, vol. 63, no. 2, pp. 277–303, 2021. [CrossRef] [Google Scholar] [Publisher Link]