Mobile Information Systems 7 (2011) 123–145
DOI 10.3233/MIS-2011-0114
IOS Press

Optimizing the performance and robustness of type-2 fuzzy group nearest-neighbor queries

Nasser Ghadiri, Ahmad Baraani-Dastjerdi*, Nasser Ghasem-Aghaee and Mohammad A. Nematbakhsh
Department of Computer Engineering, Faculty of Engineering, University of Isfahan, Isfahan, Iran

Abstract. In Group Nearest-Neighbor (GNN) queries, the goal is to find one or more points of interest with minimum sum of distance to the current location of mobile users. The classic forms of GNN use Euclidean distance measure which is not sufficient to capture other essential distance perceptions of human and the inherent uncertainty of it. To overcome this problem, an improved distance model can be used which is based on a richer, closer to real-world type-2 fuzzy logic distance model. However, large search spaces as well as the need for higher-order uncertainty management will increase the response times of such GNN queries. In this paper two fuzzy clustering methods combined with spatial tessellation are exploited to reduce the search space. Extensive evaluation of the proposed method shows improved response times compared to naïve method while maintaining a high quality of approximation. The proposed uncertainty management method also provides robustness to movement of mobile users, eliminating the need for full re-computation of candidate clusters when the locations of group members are changed.

Keywords: Group-based search, location-based service, mobile query, computational geometry, computational intelligence, context-awareness

1. Introduction

Mobility of people has enabled collaborative and group-based activities for anytime, anywhere access to information. When processing group-initiated queries in different contexts, we have to cope with inherent uncertainty emerged from various preferences and perceptions of group members. We will study on an important class of such queries after pointing out some motivating scenarios and research challenges.

1.1. Motivating scenarios

In some applications like tourism and crisis management there are situations where a group of mobile users search for nearest point(s) to meet or to perform a specific task. Group nearest neighbor (GNN) queries are defined using a source dataset of m points $P = \{p_1, p_2, \ldots, p_m\}$, i.e. a list of hotels or other destinations in a city, and a set of n query points $H = \{h_1, h_2, \ldots, h_n\}$ that represent members of a

*Corresponding author. Tel.: +98 311 793 4095; Fax: +98 311 793 2670; E-mail1: ahmadb@eng.ui.ac.ir; ahmadbaraani@yahoo.com.

group, where the problem is to find the nearest point(s) in P which minimize the sum of distances to all query points in H [30,31]. This is the main difference of GNN queries with traditional nearest-neighbor queries where only the distance to a single query point is considered. In recent years many improvements are suggested for GNN queries. However, an important aspect in some mobile applications is the role of human and the subjective satisfaction of user [13], which is often neglected.

In a recent work by Ghadiri et al. [12], it has been argued that a main problem in traditional GNN is how to model the *distance* itself. A GNN query is performed by a *group* of people. In a group, each member may have his/her own perception of distance, with possibly different preference over each type of perception. Most of the current approaches to GNN are based on a single-measure and crisp distance function, usually the Euclidean or the spatial network distance models. Such models are too simple to handle the perceptions of distance by human and the underlying higher-order uncertainties which arise in a group. In [12], the authors suggest a multi-measure distance model, called GREST (GRoup Economical, Spatial and Temporal) distance. Instead of single-measure distances like Euclidean distance, GREST is based on three important aspects of distance perception by human. Their model is based on the fact that query points are not just points, but a human is situated at each point, with multiple aspects of distance perception as well as a set of context-dependent preferences. It uses interval type-2 fuzzy sets (IT2FSs) and the linguistic weighted average reasoning method, to manage the higher-order uncertainty of group-based distance measurement as required in real-world GNN queries. Improved quality of consensus has been an important result of their work when handling the perceptions of distance and individual preferences by such model. However, the complicated distance modeling and the type-2 fuzzy reasoning method requires heavy computation, which makes the GREST-based GNN method difficult to use in practical applications such as location-dependent queries which are increasingly applied and require solid frameworks as well as high performance methods [25,33]. It should be noted that existing methods for performance improvement of GNN queries cannot be used for GREST-based queries, since GREST is a *user-dependent* distance measure. In other words, the distance from any source point h_i to destination point p_j depends on the user who is situated at the source point, because every user is allowed to use his/her own *perceptions* of distance and *preferences* over each type of distance, expressed by words.

Therefore, we need performance improvement methods for GREST-based GNN queries. Two such methods are already presented in [12] with some limitations. The first method is dividing the group members into a few sub-groups, depending on their locations. This approach is useful only with specific arrangements of member location. They also used the Minimum Bounding Rectangle (MBR) to prune the destination points which may lead to unwanted pruning of good locations in some situations.

Moreover, many existing GNN approaches only return the single *best* result, while in many situations the group is searching for a set of top-k results. Based on such requirements, we need to design a new query processing method which allows quickly finding a set of top-k results, possibly by some approximations. The fast, approximated result set should be as close as possible to the ideal set resulted from naïve, linear scan method of GREST-based GNN search in [12]. Since the mobile users may change locations while searching simultaneously, we are also interested in analyzing the robustness of the approximation method to mobility of users.

1.2. Contributions

To meet the aforementioned requirements, in this paper we suggest a novel approach for GREST-based GNN queries. It exploits two fuzzy clustering methods and spatial tessellation to divide our set of destination locations of a city into a set of regions. Executing the GNN query involves finding the best

candidate region(s), followed by searching inside those regions for top-k GREST-based GNN results. We have also extensively evaluated our approach with several indices. The main contributions of this work are:

(1) Introducing a two-level approach for GNN queries, In this two-level approach, first we gain some high level information about the distribution of destination locations by fuzzy clustering and spatial tessellation with Euclidean distance, which divides the whole area into a set of regions. Each region may contain one or more destination locations. When executing each top-k GNN query, the set of regions is searched (instead of the large set of individual destinations) using GREST-based GNN toward finding the best candidate regions which contain at least k points. After selection of candidate region(s) a finer level GREST-based GNN is performed over them returning the desired top-k destinations.
(2) Evaluating the quality of partitioning the space by fuzzy clustering with four indices.
(3) Evaluating the quality of approximation in GREST-based GNN queries, based on the Jaccard similarity measure between two interval type-2 fuzzy sets.
(4) Analyzing the effect of moving people on changes of best candidate region(s), to evaluate the advantages of our two-level approach combined with type-2 fuzzy uncertainty management for a group of mobile users.

The rest of the paper is organized as follows. Section 2 compares the related works. Section 3 overviews some basic concepts and techniques. The GREST distance model is described in Section 4. Section 5 presents our proposed methods of GREST-based GNN query processing which utilizes two types of fuzzy clustering and tessellation to find the best dominating region(s) followed by searching inside those regions using the GREST as a type-2 fuzzy multi-measure distance model. This section also presents a comprehensive set of evaluation criteria for the generalized GNN queries which use multi-measure distances like GREST. Section 6 highlights the important experimental and evaluation results of the proposed method and Section 7 concludes the paper.

2. Related works

Mobile and location-dependent queries have attracted extensive research in recent years [10,15,25]. The effect of movement of the mobile user between cells [20] and its database aspect [39] are already studied as well as nearest-neighbor queries [45] and a method for managing their uncertainty [38]. These are personal, non-group queries. The term GNN for group-based queries was originally coined by Papadias et al. [30]. They also used the term Aggregate Nearest Neighbor (ANN) for such queries [31] and examined two new aggregate functions *min* and *max*. They showed that GNN can be used in many interesting application domains like tourism, finding optimum locations for establishing chained stores and virtual battlefield. To improve the performance, they offered three methods based on the Euclidean distance model. Many performance improvement methods are suggested for GNN queries by other researchers, which are often based on single measure distances like the Euclidean distance. Two geometric pruning methods for search space reduction were also offered by Li et al. [22].

Voronoi diagrams have been a well-known method for processing mobile queries [43] and mobile navigation [44,49]. In a recent work by Sharifzadeh and Shahabi [37], Voronoi diagrams are used to improve the performance of several types of nearest neighbor queries, including GNN and top-k GNN query. Voronoi diagrams are also exploited in the specific class of spatial network queries [36,50]. In group-based case, in addition to Euclidean distance, GNN query over spatial networks is also studied [47].

Safar [35] proposed a method for processing GNN queries in spatial network databases using Voronoi diagrams. In his method, the shortest paths are pre-computed and stored in a specific data structure to be used for improving the performance of GNN query. A similar class of group-based queries namely group nearest-group query is introduced by Deng et al. [8] and its extension for querying over moving objects is studied by Hu et al. [19]. The problem of privacy in GNN query is considered by Hashem et al. [16].

To the best of knowledge of the authors, all of current studies on GNN query processing and improving its performance are based on a single-measure distance model, either Euclidean or the network model. Moreover, while fuzzy logic is used in query processing [7,21,46], and the role of user and collaboration is also considered [6,18], none of the aforementioned methods handles the high-order uncertainty. A probabilistic approach to uncertainty management for GNN queries was proposed by Lian et al. [23] which uses the Euclidean distance. The probabilistic approach cannot be used to model human's perception of distance. In contrast, a type-2 fuzzy distance model named GREST [12] makes it possible for mobile users to have multiple types of distance perception with different perceptions toward each type, all described by natural language words. It has been shown that the GREST distance model provides much higher degrees of consensus to satisfy as many group members as possible, and is further improved by sub grouping [12]. However, the GNN query based on the GREST distance model requires heavy computations which prevent its efficient use real-world mobile usage scenarios.

Existing methods of GNN performance improvement are not applicable to the GREST distance model, as its computation depends on each user's perception of each distance type. We cannot build a single index as required by most performance improvement methods. This paper is an effort to overcome this problem of GREST-based GNN queries by using fuzzy clustering and spatial tessellation. The main difference between this paper and our previous work [12] is in their scope. It was very different and focused on human-centric aspects of the proposed GREST distance model, such as the quality of group consensus and sub-grouping. This paper focuses on performance improvement of GREST-based GNN queries and providing more robustness to changes in mobile user locations that use the same distance model.

3. Preliminaries

Three topics are covered by this section. First, we give a brief overview of type-2 fuzzy sets that are used for handling high-order uncertainty in the GREST distance model. Section 3.2 introduces two fuzzy clustering methods which are used in our approach to partition the search space into a set of smaller regions. Voronoi diagrams are explained in Section 3.3 and will be used later to convert the fuzzy partitions into hard ones.

3.1. Modeling and reasoning in type-2 fuzzy systems

Real-world concepts like distance are inherently uncertain. For example, there is no exact and generally accepted definition of the word *near*. In fact, different people may have different beliefs about words. Although fuzzy sets are closer to the real-world than crisp sets, a key problem in some situations is how to define the membership functions for the classic fuzzy sets. Mendel has shown that two forms of *higher-order* uncertainties exist in such cases [26]. The first type is *intra-uncertainty.* When we ask a user to define the shape of a fuzzy set, he/she gives only an uncertain definition of its boundaries. The

second type is *inter-uncertainty.* Different people may have different interpretations of how a fuzzy set is exactly defined.

Interval type-2 fuzzy sets (IT2FS) are proposed in [29] to handle such higher-order uncertainties. The IT2FSs will help us to model both the intra-uncertainty and inter-uncertainty about a word, using a *footprint of uncertainty* (*FOU*) [27], which can be considered as a fuzzy membership function with an extra degree of freedom. The GREST distance model which is introduced in Section 4 uses the FOUs for every distance measure such as spatial, temporal and economic type. Moreover, different people may have different preferences toward each type of distance. The distance model uses IT2FSs again to allow people to use a predefined set of words (e.g. *small, medium, large*) to express their preferences toward spatial, temporal and economical distance types in natural language.

3.2. Fuzzy clustering methods

As the first step of partitioning the large space of GNN queries, we will use fuzzy clustering. In clustering, objects (destination locations in this context) are grouped together according to a similarity measure. This helps us to find the underlying structure in our data. Knowing such structure can potentially improve our GNN query processing methods. In literature, fuzzy partitioning has shown two advantages over crisp or hard clustering methods like K-Means and K-Medoids [34]: First, it allows each object to be a member of more than one cluster, using fuzzy membership functions. Second, it prevents the problem of local minima as an optimization problem. For the purpose of our work, we use the second feature to partition the search space efficiently.

The most widely used method of fuzzy clustering is Fuzzy C-Means (FCM) [5]. The algorithm is based on minimizing an objective function. The number of clusters is determined initially by the user. Distance measurement in FCM is based on a dissimilarity function which computes the distance from each point to each cluster center using the Euclidean distance measure. Based on such distance measurement, FCM generally performs well in finding 'globular' clusters.

An alternate of FCM algorithm is known as GK clustering proposed by Gustafson and Kessel [1]. The main difference between GK and FCM is in their dissimilarity functions. Each cluster in GK is allowed to have its own norm inducing matrix for distance measurement. For example, distance measurement in horizontal direction toward the center of a specific cluster may assume a higher weight than the vertical direction and vice versa. This allows the detection of 'non-globular' clusters by computing distance on different scales for each orientation. We expect this feature of GK clustering to be practically useful, since the Points of Interest (POIs) in our GNN queries are not necessarily distributed in globular regions. For instance, the POIs alongside a river may form a straight, non-globular cluster.

After performing fuzzy clustering, we will convert soft partitions to hard ones for making decisions about the candidate points to carry out the GREST-based GNN search. One way of doing this is to assign each input to the cluster with the highest degree of membership. Another way is to augment the clustering with spatial tessellation which is introduced in the next subsection.

3.3. Spatial tessellation

In the second step of our heuristic in partitioning the large space of GNN queries we will use spatial tessellation. Spatial tessellation by using Voronoi diagram is a well-known method in computational geometry for partitioning the space. Given a set of *generator* points, the ordinary *Voronoi polygon* associated with every generator point p_i in n-dimensional space is defined as $V(p_i)$ shown by Eq. (1) [9]:

$$V(p_i) = \{x \in \mathbb{R}^n | d_S(x, p_i) \leqslant d_S(x, p_j) \text{ for any } j \neq i\} \tag{1}$$

where d_S denotes the ordinary spatial (Euclidean) distance. A Voronoi diagram divides the given space into several areas. Each area contains all points which are closer to p_i than any other $p_{j,}$ for every $j \neq i$. In Fig. 2 parts (a2) and (b2) two examples of Voronoi diagrams can be seen with two different sets of 25 generators each. To partition the space using Voronoi diagrams, the set of generator points must be supplied by user. For our specific GNN application, we will use cluster centers resulted from the fuzzy clustering as generators.

By using tessellation after clustering, we expect better formation of hard clusters from soft clusters created by FCM or GK clustering methods. This method of enhancing the boundaries of fuzzy partitions was first introduced by Hoppner and Klawonn [17] and we use it in our GNN processing scheme. In literature, Voronoi diagrams are also used for detecting proximity relationships between clusters faster than naïve method of cluster-by-cluster comparison [4]. Comparing Voronoi diagrams to clustering, we believe that each partitioning method has its own strengths and weaknesses. Our idea will be based on incorporating the strong points of both methods to prepare the search space for complex uncertainty-aware search based on a type-2 fuzzy logic distance model.

4. Overview of the GREST distance model

This section contains an overview of the important features of the GREST distance model introduced by the authors of [12]. The GREST distance model replaces Euclidean distance of classic GNN queries and, because of being user-dependent, prevents us from using classic methods of performance improvement for GNN queries.

As discussed in Section 1.1, the Euclidean distance which is widely used in classic GNN queries has some shortcomings for real-world applications of such queries. The 'G' in GNN stands for 'group' and the members of any group may have different types of distance perception. Moreover, their *preferences* toward each type of distance measurement can be different. When trying to take such aspects into account, we will also need to cope with to another important problem, the problem of higher-order uncertainties. The GREST distance model [12] provides a solution for these problems and we give a brief overview of its features here.

The first strong point of GREST is that different types of distance perception are allowed. In addition to the *spatial* measure which is represented by the Euclidean distance in literature, there are two other important aspects of distance measurement by human, namely *temporal* and *economical.* For example:

- Using the *spatial* measure, destinations are considered *near* to some people if they are within a radius of, say, 500 meters from their current location.
- In a *temporal* context, distance is proportional to the travel time between any two points. Again, the word *near* can be assigned to those locations which are within 15 minutes from the current location.
- From an *economic* point of view, the distance is proportional to the cost of traveling from source to destination point.

The above three common aspects of distance measurement are included in and aggregated model of distance measurement named GREST [12].

The second important capability of GREST is coping with higher-order uncertainty. It has been argued that different members of a group may have different opinions about the boundaries of given words for distance perception and for preferences [12]. For example, a member h_1 may use the word *near* in a spatial context for those locations which have a distance of [0.5,1.0] kilometers, while another member h_2 may define the same word as the [1.0, 2.0] interval. The same problem exists for expressing the

preferences by different members. Again, if we ask the members to define an interval between 0-1 for their preferences such as *small, moderate, large*, a member like h_1 may define the word *small* as [0.1–0.3] and another member h_2 may define the same word as [0.15–0.25]. The authors in [12] show that such complexities give rise to higher-order uncertainties which cannot be handled by current methods. To solve the aforementioned problems, they have introduced the GREST distance model which allows the members of any group to define and use a set of words in natural language to express each type of their perception of distance (e.g. *near* for the spatial and *moderate* for the economical perception). They can also use words like *small* and *large* to express their preferences toward each distance type. To cope with higher-order uncertainties of distance perception, GREST uses IT2FS for each distance measure and the Linguistic Weighted Average (LWA) as introduced in [28,40] for their combination. The GREST function is shown by Eq. (2).

$$\tilde{D}_{GREST}(h,p) = \frac{\tilde{W}_S \tilde{D}_S(h,p) + \tilde{W}_T \tilde{D}_T(h,p) + \tilde{W}_E \tilde{D}_E(h,p)}{\tilde{W}_S + \tilde{W}_T + \tilde{W}_E} \tag{2}$$

where $\tilde{D}_S(h,p)$, $\tilde{D}_T(h,p)$, $\tilde{D}_E(h,p)$ represent the perceptions of spatial, temporal and economical distance by each member respectively. The preferences of this member over each distance type are defined as words represented by and $\tilde{W}_S$, $\tilde{W}_T$, $\tilde{W}_E$ for spatial, temporal and economical preferences respectively. For example a user who prefers temporal distance over spatial and economic distances will be able to assign type-2 fuzzy weights *small*, *large*, *small* to $\tilde{W}_S$, $\tilde{W}_T$, $\tilde{W}_E$ respectively. The $\tilde{D}_{GREST}(h,p)$ represents the distance from any member h to any destination p.

It should be noted that all distance measures ($\tilde{D}_S$, $\tilde{D}_T$, $\tilde{D}_E$) and their weights ($\tilde{W}_S$, $\tilde{W}_T$, $\tilde{W}_E$) in Eq. (2) are IT2FSs represented by FOUs and this equation is for illustration purpose. When processing a GNN query, $\tilde{D}_{GREST}(h,p)$ will be computed for every user and every destination. The results of such personal distance perceptions are integrated at the next level of reasoning, with the possibility of assigning different weights to different users.

The GREST distance function shown by Eq. (2) have been used in GREST-based GNN queries [11] and in our process as described in the next section.

5. The proposed method for GREST-based GNN query processing

In Section 5.1 we present the proposed methods of performance improvement for GREST-based GNN queries. Section 5.2 provides several evaluation criteria which will be used in our experiments to evaluate the proposed methods.

5.1. Overview of the process

As discussed in Section 2, existing GNN performance improvement and search space pruning methods are distance-specific (usually Euclidean or network distance) and cannot be used with a multi-measure, user-dependent distance model like GREST. However, the existing performance improvement methods for GNN queries are either *index-based* or use *pre-computation*. In several query processing schemes, pre-computation methods have shown better performance [23,32]. Our strategy is also based on partitioning the space as a pre-computation phase which is performed only once for each combination of a city, its POI locations, and each configuration of group preferences. When executing the GREST-based GNN query, this partitioned space makes it possible to begin the search at a coarse level toward finding the

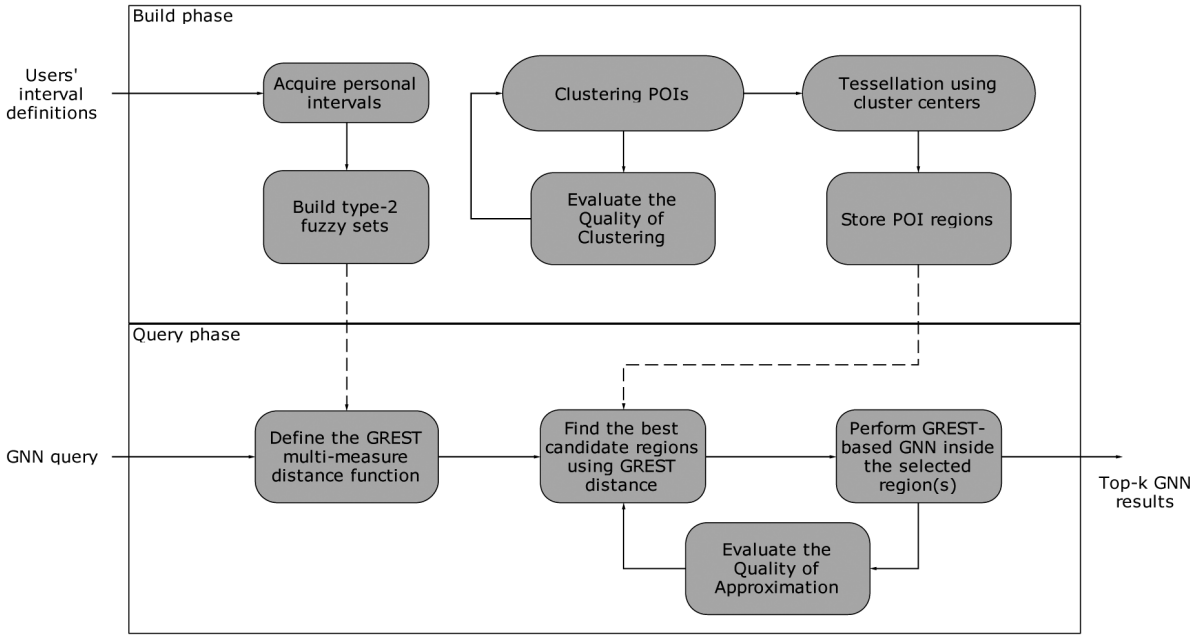

Fig. 1. Overview of the proposed GNN query processing method.

best approximate region, and performing the GNN at a finer level inside that region. We can divide the process into two phases, as shown in Fig. 1.

The *build phase* comprises two important activities. The first activity is building the interval type-2 fuzzy sets for desired types of distance measurement. Type-2 fuzzy FOUs are constructed from user-defined intervals using the Interval Approach introduced by Liu and Mendel [24]. The input to this activity is the set of user-defined intervals for each fuzzy linguistic variable. The outputs are the type-2 fuzzy FOUs which reflect the *group's* definition of the fuzzy sets involved in distance perceptions and preferences. The resulting FOUs are stored and will be used at query phase. The second activity in the build phase is partitioning the POI space which is done by clustering the POI space to find the cluster centers and membership degrees. Then cluster centers are used as generator points for building the Voronoi diagram which determines the final partitioning of the POI space. Building the type-2 fuzzy sets is performed once for each configuration of group preferences. The whole partitioning is performed only once for each city.

In the *query phase*, first the GREST function is defined using the FOUs as already stored at build phase. This multi-measure distance function is fully customized to group's definition of uncertainty boundaries for each fuzzy set. It also provides enough flexibility to handle the type-2 fuzzy *spatial*, *temporal* and *economical* preferences of the users as weights of the linguistic weighted average. At the query phase, the stored information about space partitions are the first set of target points for GNN. To respond to the GNN query, the best candidate partitions are selected based on their distance. The GREST distance is computed for each partition using the FOUs of distances and preferences. The output of this step is a FOU itself, for which the type-2 fuzzy centroid is calculated and used as a real number to rank different partitions or POIs. This leads to selection of one or more partitions which potentially contain the desired GNN points, using the group's specific GREST function.

Two evaluation activities (one for each phase) are also used for monitoring the *quality of clustering*

and the *quality of approximation* as feedback mechanisms. Measuring the quality of clustering helps to select better values for initial number of clusters and will be described in Section 5.2.3.

To ensure the validity of the proposed optimization method, we measure the similarity between the ideal set of top-k results as retuned by the original method [12], with our optimized (approximate) method. We name it the *quality of approximation* and will describe it in Section 5.2.1. A value of 1.0 for this measure shows that all returned top-k results match the original results. Otherwise, some resulting POIs are near-ideal points.

If the first partition with the highest rank contains less POIs than requested k points, another partition will be added to the list, until the selected partitions contain at least k POIs. Finally, we perform the GNN at a finer level using the GREST function again, over all POIs of the selected partition(s) and return the top-k best GNN results.

We have two reasons for using the Euclidean distance in the pre-computation phase, in lieu of the GREST distance. First, the GREST model is necessary only when a considerable amount of uncertainty exists. When partitioning a city into a set of clusters, the distance from any point to its cluster center is rather small, and there is no need to differentiate spatial, temporal and economic measures with uncertainty. The second reason is that the GREST distance is user-dependent and even if we were able to find a clustering method which uses this model, it would be very difficult and computationally-inefficient to perform the clustering based on GREST. Therefore, we used the Euclidean distance in the pre-computation phase and the GREST distance in both levels of the execution phase.

5.2. Evaluation criteria

In this section we give an overview of several evaluation criteria which will be used to analyze the proposed method from different aspects. The main aspect is measuring the validity of our proposed method as a similarity measure between ideal and approximated top-k result sets. In addition, we need to evaluate the quality of clustering method, its effect on performance improvement and robustness to changes in user locations.

5.2.1. Quality of approximation based on IT2FS Jaccard similarity

The proposed method of processing top-k GREST-based GNN queries in Section 5.1 limits the search space to the best selected partition(s) only. Although the partition selection is based on group's customized GREST multi-measure distance, there is no guaranty that the selected partition(s) will contain all top-k results of the normal, non-partitioned execution of the naïve GREST-based GNN algorithm. In other words, we may obtain an approximate result. An important measure here is the degree to which this ranked approximate top-k set of POIs is similar to the optimum set. For this purpose, we suggest a pair-wise comparison of similarity between optimum set of POIs and the POIs returned by our method. To determine the degree of similarity between two POIs according to group's GREST function, we need to compare the output of ranking mechanism when triggered by each of them as input. We have two choices:

- Comparing their numeric ranks, which was calculated in Section 5 as the *centroid* of interval type-2 fuzzy output (FOU).
- Comparing the *FOUs* resulted from each method, using a type-2 fuzzy similarity measure.

The first option is reasonable for ranking, but may not perform well for similarity measurement. Two different FOUs may have almost equal centroids, and two similar FOUs may have different centroids due to their differences in shape. Therefore, the second option is more reasonable for similarity measurement

since it directly compares two FOUs using a type-2 fuzzy similarity measure. Several similarity measures for interval type-2 fuzzy sets are studied and compared by Wu and Mendel [41]. Their own Jaccard similarity measure for interval type-2 fuzzy sets has shown the best characteristics and we will use it here, shown by Eq. (3):

$$S_J(\tilde{A}, \tilde{B}) = \frac{p(\tilde{A} \cap \tilde{B})}{p(\tilde{A} \cup \tilde{B})} \tag{3}$$

In Eq. (3), $\tilde{A}$ and $\tilde{B}$ represent the ideal and approximated query results, and S_J shows the degree of similarity between these two sets and the maximum value of this index is 1.0 which happens when two FOUs are identical or fully similar.

We will use the average of such pair-wise measures, between the FOUs resulted from our method and those resulted from the ideal ones, for evaluating the quality of approximation in different configurations.

5.2.2. Performance improvement

Comparing the query *response time* of proposed method with previous GREST-based GNN methods is an important criterion and its calculation is straightforward. We will compute the build time and execution time of our proposed methods for each combination of clustering methods, number of clusters, with or without tessellation and with two different datasets. We define speed-up factors to show how much our method performs faster, compared to the naïve linear scan method of computing GREST-based GNN queries. They will be calculated using build time and running time measured in seconds and show the amount of time required for each phase of our method described in Section 5. One factor assumes build phase followed by a single execution of query, another assumes the more realistic case of build-once/execute many times scenario of GREST-based GNN processing.

5.2.3. Quality of clustering

Several indices are proposed in literature for evaluating the quality of clustering. No single criterion provides the 'best' information in all situations. Thus, we will use four of the evaluation methods and integrate the results to make better decisions about the clustering method and the initial number of clusters. Our clustering evaluation methods are defined as [2]:

(i) The Partition Coefficient (PC) [5], which calculates the overlapping between fuzzy partitions. The highest amount of PC shows the best clustering quality.
(ii) The Classification Entropy (CE) [5] is also similar to PC, but it calculates the fuzziness of the partitions.
(iii) The Partition Index (SC) [3], is useful when comparing several clustering methods by a fixed number of clusters. The clustering method which gives lower SC is better.
(iv) The Xie and Beni's index (XB) [42], determines both the *internal* variation of clusters and how well the clusters are separated *externally*. The best number of clusters is when the lowest value of XB is obtained.

There are also other indices like Dunn's Index and Alternate Dunn's Index but we did not get any interpretable results for our GNN context by using them.

5.2.4. Robustness (of candidate cluster) to mobility

Based on Zadeh's principle of "*don't need*" in handling uncertainty [48] and by utilizing advanced uncertainty management methods like type-2 fuzzy logic, we are interested in taking the advantage

of its robustness to small changes in user locations. For this purpose, we will simulate the result for movement of each user in either horizontal or vertical direction by various distances. There have been studies on mobility patterns which are out of the scope of this paper. However, we focus on the result of mobility regardless of its pattern, i.e. we focus on the distance traveled from original location to new random location by each member of the group. The variations on best dominating clusters are monitored. Fewer changes in best clusters will reduce the need to repeated calculation of the best clusters for local movements and will result in greater robustness to mobility of the group. After selecting the best cluster, even if it remains unchanged, a new direct GREST-based GNN on POIs inside that cluster using the GREST distance measure is still necessary, since the aggregated distances may have been changed.

6. Experimental results

This section begins with an overview of the software environment, data and configuration of preferences and distance measures. Then each set of evaluations will be presented. Our experiments performed in MATLAB, using a clustering toolbox[1] which implements several algorithms including FCM and a numerically robust version of GK clustering algorithm [14], as well as their evaluation methods. We also used IT2FS software[2] which implements interval type-2 fuzzy set algorithms including the LWA and Jaccard similarity measure. All experiments were run on a dual-core machine with 2GB of RAM.

For POI data, we extracted the location information of 1263 hotels in Paris and 195 hotels in Vienna from GeoNames.[3] They are selected as representatives of a large- and a medium-size city respectively. The location data was first imported into PostGIS and then used in MATLAB after conversion and normalization. The distribution of preferences toward three distance measures between group members is random, i.e. one-third of members prefer *spatial* measure over *temporal* and *economical* measures and so on. For POIs, random distribution over the whole city area is used. Several other distributions were already examined [11] , including random distribution over a smaller area and clustered distribution of people in 3 or 4 subgroups. However, for the purpose of this paper, the random distribution of POI over large area with random preferences is selected as a challenging case for performance improvement.

6.1. Clustering and tessellation

The results of clustering are shown in Fig. 2 (a1) for FCM and in Fig. 2 (b1) for GK clustering of Paris data for $c = 25$ clusters. The cluster centers are then used as generator points to perform the tessellation by Voronoi diagram as shown in Fig. 2 parts (a2) and (b2). The overlay of clustering with Voronoi diagram is also shown in Fig. 2 parts (a3) and (b3). As can be seen from the Fig. 2, the shape of FCM clusters is almost globular, while some GK clusters have different, non-globular shapes. Since the natural grouping of POIs, i.e. city regions, is rarely globular, the GK clustering detects them better than FCM. However, GK also produces some unwanted, interfering irregularities on cluster boundaries which we try to modify at the next step of our process by the help of Voronoi polygons.

The Voronoi polygons also follow cluster boundaries. This can be seen by comparing the shared boundary at empty area on Fig. 2 (a3) at the coordinates of (0.8,0.7) where cluster boundaries and

[1] http://www.fmt.vein.hu/softcomp/.
[2] http://sipi.usc.edu/~mendel/software/.
[3] http://www.geonames.org/data-sources.html.

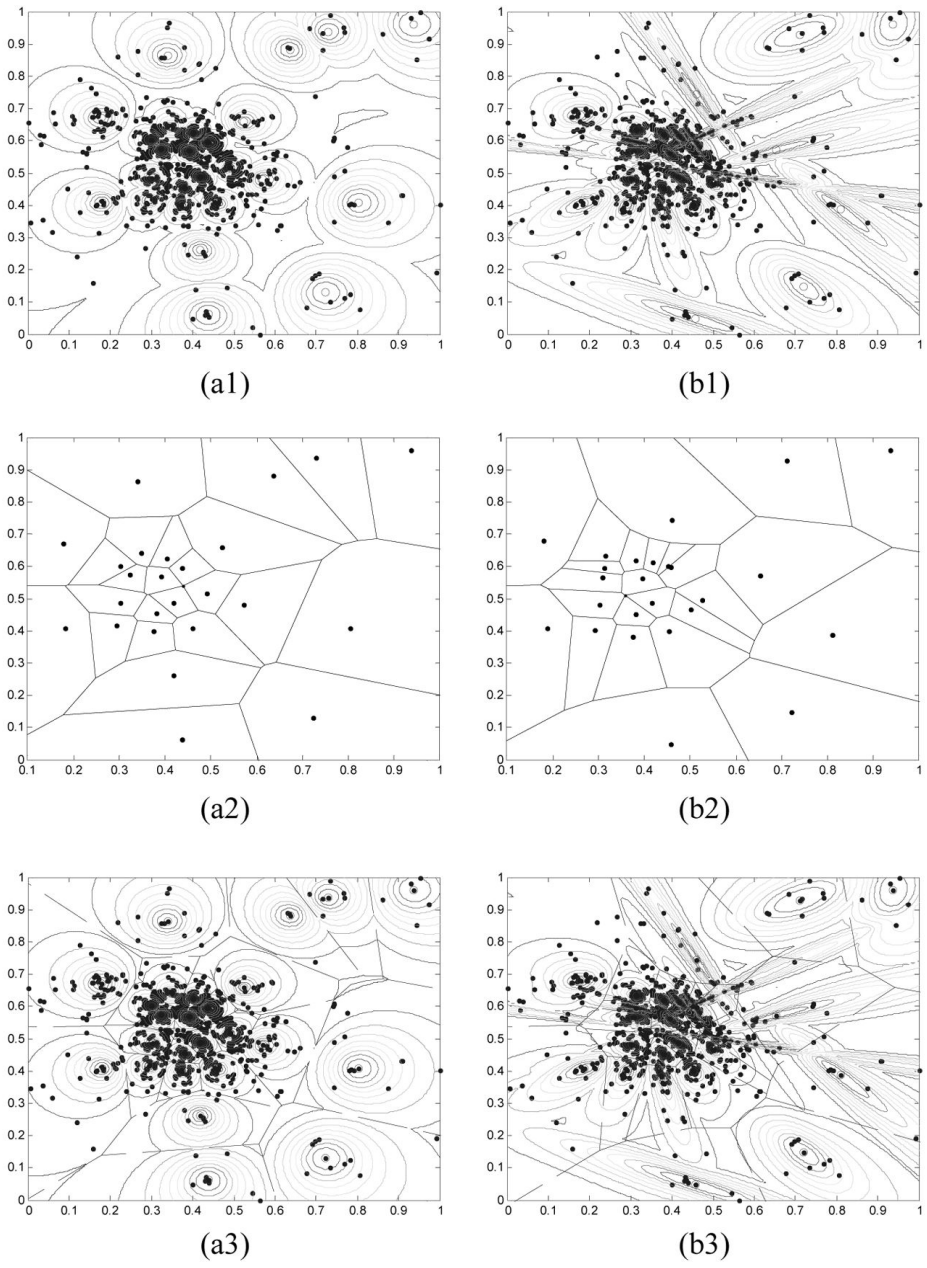

Fig. 2. Partitioning by FCM clustering (a1) and GK clustering (b1), followed by tessellation (a2,b2) and overlay of the clustering and tessellation (a3,b3). Initial number of clusters is $c = \mathbf{25.}$

Voronoi shared edge overlap at the same location. Moreover, it can be observed intuitively from Fig. 2 parts (b1)–(b3) that Voronoi tessellation has the effect of making irregular boundaries more uniform. This is the key result of performing Voronoi tessellation over cluster centers. Considering the lower GK clusters in Fig. 2 (b1), the unwanted protrusion of clusters into each other is clear. The corresponding

Voronoi polygons modify those boundaries toward a more realistic natural partitioning.

As a result, many central regions in Fig. 2 (b2) have well-formed rectangular shapes which conforms to actual shape of some municipal regions.

In other words, the Mahalanobis distance in GK clustering allows a better optimization process for finding the centers of natural clusters, and the uniform Euclidean distance measure of Voronoi diagram modified the boundaries of resulting regions.

6.2. Search performance

We evaluated the performance improvement of our proposed methods according to the criteria in Section 5.2. Both FCM and GK clustering methods were applied with and without Voronoi tessellation over two datasets. Cluster size plays a crucial role here, since the selected best cluster(s) at the first step of our GREST-based GNN query processing method (see Section 5) must have two important, yet contradictory, characteristics:

- The best cluster must contain *as much ideal points as possible*, to preserve the quality of approximation and good 'resembling' of the ideal set by the actual results.
- This cluster should be *as compact as possible*, to prevent long response times as a result of processing more points than required.

Therefore, we expect more performance improvement from the high quality clusters without any loss of the quality of approximation. We performed our experiments with several cluster sizes selected from previous experiments in Section 6.1.

Table 1 shows the details of this set of experiments with different methods and cluster sizes. In this table, c is the number of clusters, N_B shows the number of best dominating cluster(s) which contain at least k points and N_{POI} is the actual number of POIs in that cluster(s). We selected $k = 12$ assuming that top-12 points are requested by GREST-based GNN query. The columns T_{BUILD} and T_{RUN} show the build time and running time in seconds.

S_{R1} and S_{R10} are the speed-up factors calculated as the ratio of response time by our method to the response time of naïve method for each dataset. J_X is the average quality of approximation for each row. The '-V' in 'method' column means using Voronoi tessellation.

Considering the N_B column, it can be observed that except for the GK clustering for Paris with $c = 75$, one cluster has been sufficient in other cases. This is a weakness of GK with this cluster size. However, the interesting point is that Voronoi tessellation contributes very well to solving this problem by allowing this cluster to contain 19 points in the area surrounded by its Voronoi polygon.

For the Paris dataset, GK clustering with $c = 15$ combined with Voronoi tessellation outperforms other methods by running 9.9x faster for single-run and 11.8x for 10-run successive query executions. However, the quality of approximation has reached its maximum value by using the same method of GK-V, with cluster size of $c = 75$. At this cluster size, speed-up factors are still reasonable (3.6 and 10.7 for single- and 10-run respectively).

For the experiments on the Vienna dataset, shown on the lower part of Table 1, best performance improvements are achieved by GK clustering again. While for single-run case the FCM clustering with Voronoi (FCM-V) reached a 4.4x speed-up factor, GK clustering gives 5.6x at the cluster size of $c = 20$ and GK clustering with Voronoi gives 5.6x at $c = 10$, though with slightly lower quality of approximation. While we observed small changes in build times and running times between successive executions, the order of speed-up factor and the quality of approximation remain unchanged.

Table 1
Results of evaluating the performance improvement of our proposed method

Dataset	*Method*	c	N_B	N_{POI}	T_{BUILD}	T_{RUN}	S_{R1}	S_{R10}	J_X
Paris	FCM	20	1	108	1.85	9.26	8.3	9.8	0.7821
		40	1	85	1.38	8.93	9.0	10.2	0.7634
		55	1	58	2.30	8.09	8.9	11.1	0.7634
		75	1	57	2.56	9.46	7.7	9.5	0.7634
		90	1	59	3.97	10.68	6.3	8.4	0.7634
	FCM-V	20	1	114	1.60	9.55	8.3	9.5	0.8384
		40	1	85	1.32	8.97	9.0	10.2	0.7821
		55	1	59	2.37	8.21	8.8	11.0	0.7821
		75	1	56	2.69	9.43	7.6	9.5	0.7821
		90	1	58	3.94	10.62	6.4	8.4	0.7821
	GK	15	1	103	1.87	8.58	8.9	10.6	0.7428
		30	1	103	3.34	9.50	7.2	9.4	0.8775
		60	1	57	24.61	8.36	2.8	8.6	0.7634
		75	2	61	19.24	9.71	3.2	8.0	0.7625
		90	1	46	35.40	9.72	2.1	7.0	0.7623
	GK-V	15	1	92	1.65	7.67	**9.9**	**11.8**	0.7930
		30	1	102	2.94	9.45	7.5	9.5	0.8388
		60	1	60	21.24	8.59	3.1	8.6	0.7634
		75	1	19	19.27	6.75	3.6	10.7	**0.9924**
		90	1	46	38.73	9.78	1.9	6.8	0.7821
Vienna	FCM	10	1	36	0.77	3.36	3.5	4.1	0.9988
		20	1	28	0.58	3.40	3.6	4.1	0.9988
		40	1	14	0.79	3.84	3.1	3.6	0.9988
		50	1	17	0.93	4.78	2.5	2.9	0.9988
	FCM-V	10	1	29	0.43	2.79	**4.4**	5.0	0.9988
		20	1	26	0.51	3.27	3.8	4.3	0.9988
		40	1	14	0.84	3.87	3.0	3.6	0.9988
		50	1	16	0.97	4.72	2.5	3.0	0.9988
	GK	10	1	45	1.15	3.88	2.8	3.6	0.9988
		20	1	13	1.56	2.40	3.6	**5.6**	0.9631
		40	1	14	2.54	3.84	2.2	3.5	0.9988
		50	1	14	3.38	4.59	1.8	2.9	0.9988
	GK-V	10	1	24	1.12	2.42	4.0	**5.6**	0.9789
		20	1	21	1.56	2.94	3.2	4.6	0.9965
		40	1	15	2.57	3.92	2.2	3.4	0.9988
		50	1	15	3.39	4.64	1.8	2.9	0.9988

A graphic representation of the performance improvement is depicted in Fig. 3. For the Paris dataset, although FCM exhibits higher speed-up factor for some cases in Fig. 3(a), its quality of approximation is low, compared to GK clustering combined with Voronoi which offers the best combination of performance improvement and quality of approximation. For the Vienna dataset shown in Fig. 3(b), there is a difference between FCM and GK, with GK being slightly better in terms of speed-up factor. The diagrams also clearly show that with increasing number of clusters, performance will not increase due to the loss of clustering quality and increased time for cluster search.

6.3. Quality of approximation

This part of experiment shows the validity of our proposed approximation method. We already observed the average similarity measure between interval type-2 fuzzy sets of the ideal and actual POI sets in the last column of Table 1. This measure was calculated as the average of individual similarity

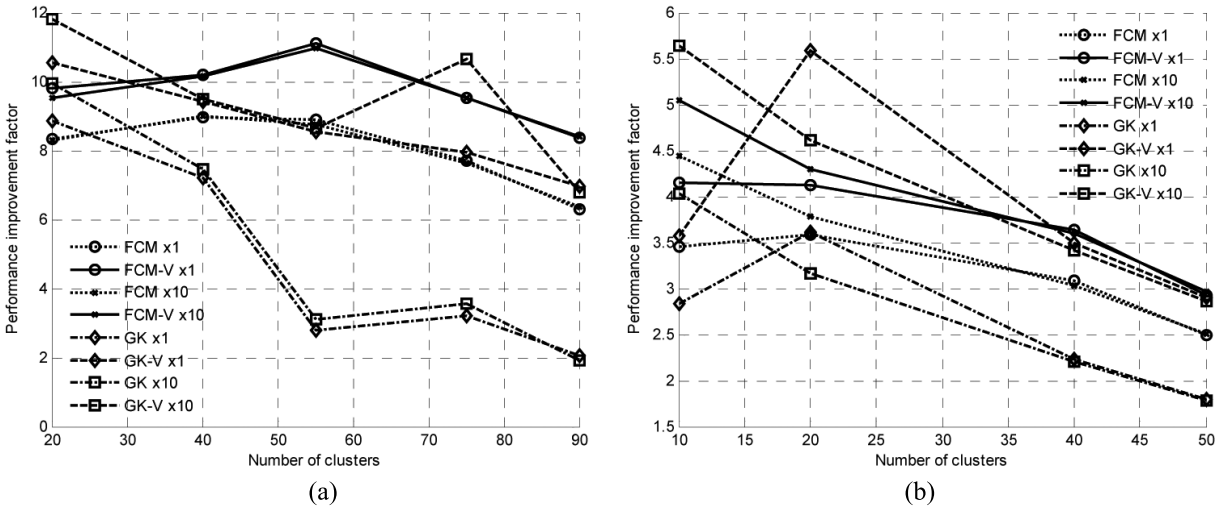

Fig. 3. Performance improvement results (a) for Paris dataset (b) for Vienna dataset.

values between any two POIs from the two sets, as described in Section 5.2.1. Table 2 shows the detailed similarity measures for selected configurations which dominated in one or more aspects in Table 1.

The columns $I1..I12$ represent the FOUs of the ideal k points as calculated by the naïve method. The rows $A1.A12$ represent the FOUs of the actual k points as calculated by our method. For the Paris dataset, when using GK clustering in combination with Voronoi at $c = 15$, the table shows that only the first 3 points $A1, A2, A3$ resulted from our method are similar to the ideal points with a similarity degree of above 0.9. Other points are similar to a degree of about 0.7, which leads to the average approximation quality of 0.7930. When using GK clustering and Voronoi at $c = 75$, we observe strong similarity between the actual and ideal sets. Many points are similar with similarity degree of 1.0. In fact, they are either identical points or points that are very close to each other, based on the GREST distance measure. The result is an average approximation quality of 0.9924 which is very interesting. Recall from Section 6.1 that $c = 75$ was one of the best values in quality assessment for GK method.

For the Vienna dataset, many similarity values are either 1.0 or very close to 1.0, with FCM-V being slightly better than GK-V in some cases. An interpretation is that cities with smaller number of POIs are less sensitive to initial number of clusters. In such cases, both FCM and GK clustering in combination with Voronoi tessellation provide good approximation quality.

6.4. The effect of mobility

As discussed in Section 5.2, by exploiting the advanced uncertainty management methods like type-2 fuzzy logic, we are interested in taking the advantage of its robustness to small changes over the input domain. For this purpose, we simulated the movement of each member of the group, in either horizontal or vertical direction, by various distances from 250 meters to 2000 meters. Table 3 shows the result of this experiment for the same important configurations that were examined earlier from other aspects. In this table, N is the order of the best clusters. The columns I and R show the cluster number and its ranking value respectively. The ranking value is the centroid of the IT2FS resulted from calculating the GREST to each cluster by the LWA method according to Section 5.1. For the GK clustering over the Paris dataset at $c = 15$ with Voronoi tessellation where the approximation quality was not very high,

Table 2
Results of evaluating the quality of approximation using the Jaccard similarity measure between type-2 fuzzy sets of ideal and actual selections

Configgfig	A/ I	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10	I11	I12
GK-V	A1	0.9894	0.9894	0.9942	0.9961	0.9961	0.9961	0.9961	0.9961	0.9961	1.0000	0.9902	0.9889
Paris	A2	0.9075	0.9075	0.9117	0.9135	0.9135	0.9135	0.9135	0.9135	0.9135	0.9170	0.9188	0.9197
$c = 15$	A3	0.9021	0.9021	0.9063	0.9081	0.9081	0.9081	0.9081	0.9081	0.9081	0.9115	0.9184	0.9194
	A4	0.7523	0.7523	0.7556	0.7572	0.7572	0.7572	0.7572	0.7572	0.7572	0.7599	0.7666	0.7676
	A5	0.7523	0.7523	0.7556	0.7572	0.7572	0.7572	0.7572	0.7572	0.7572	0.7599	0.7666	0.7676
	A6	0.7383	0.7383	0.7415	0.7431	0.7431	0.7431	0.7431	0.7431	0.7431	0.7458	0.7522	0.7532
	A7	0.7383	0.7383	0.7415	0.7431	0.7431	0.7431	0.7431	0.7431	0.7431	0.7458	0.7522	0.7532
	A8	0.7357	0.7357	0.7389	0.7405	0.7405	0.7405	0.7405	0.7405	0.7405	0.7431	0.7495	0.7505
	A9	0.7357	0.7357	0.7389	0.7405	0.7405	0.7405	0.7405	0.7405	0.7405	0.7431	0.7495	0.7505
	A10	0.7357	0.7357	0.7389	0.7405	0.7405	0.7405	0.7405	0.7405	0.7405	0.7431	0.7495	0.7505
	A11	0.7343	0.7343	0.7375	0.7391	0.7391	0.7391	0.7391	0.7391	0.7391	0.7417	0.7481	0.7491
	A12	0.7250	0.7250	0.7282	0.7298	0.7298	0.7298	0.7298	0.7298	0.7298	0.7324	0.7387	0.7397
GK-V	A1	**1.0000**	**1.0000**	0.9951	0.9933	0.9933	0.9933	0.9933	0.9933	0.9933	0.9894	0.9797	0.9784
Paris	A2	**1.0000**	**1.0000**	0.9951	0.9933	0.9933	0.9933	0.9933	0.9933	0.9933	0.9894	0.9797	0.9784
$c = 75$	A3	0.9933	0.9933	0.9960	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9961	0.9863	0.9850
	A4	0.9933	0.9933	0.9960	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9961	0.9863	0.9850
	A5	0.9933	0.9933	0.9960	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9961	0.9863	0.9850
	A6	0.9933	0.9933	0.9960	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9961	0.9863	0.9850
	A7	0.9933	0.9933	0.9960	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9961	0.9863	0.9850
	A8	0.9933	0.9933	0.9960	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9961	0.9863	0.9850
	A9	0.9894	0.9894	0.9942	0.9961	0.9961	0.9961	0.9961	0.9961	0.9961	**1.0000**	0.9902	0.9889
	A10	0.9797	0.9797	0.9844	0.9863	0.9863	0.9863	0.9863	0.9863	0.9863	0.9902	**1.0000**	0.9986
	A11	0.9784	0.9784	0.9831	0.9850	0.9850	0.9850	0.9850	0.9850	0.9850	0.9889	0.9986	**1.0000**
	A12	0.9741	0.9741	0.9788	0.9807	0.9807	0.9807	0.9807	0.9807	0.9807	0.9845	0.9943	0.9956
FCM-V	A1	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9957	0.9957
Vienna	A2	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9957	0.9957
$c = 10$	A3	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9957	0.9957
	A4	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9957	0.9957
	A5	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9957	0.9957
	A6	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9957	0.9957
	A7	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9957	0.9957
	A8	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9957	0.9957
	A9	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9957	0.9957
	A10	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9957	0.9957
	A11	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	**1.0000**	**1.0000**
	A12	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	**1.0000**	**1.0000**
GK-V	A1	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9957	0.9957
Vienna	A2	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9957	0.9957
$c =10$	A3	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	**1.0000**	**1.0000**
	A4	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	**1.0000**	**1.0000**
	A5	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	**1.0000**	**1.0000**
	A6	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	0.9957	**1.0000**	**1.0000**
	A7	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9653	0.9653
	A8	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9653	0.9653
	A9	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9653	0.9653
	A10	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9612	0.9653	0.9653
	A11	0.9569	0.9569	0.9569	0.9569	0.9569	0.9569	0.9569	0.9569	0.9569	0.9569	0.9611	0.9611
	A12	0.9569	0.9569	0.9569	0.9569	0.9569	0.9569	0.9569	0.9569	0.9569	0.9569	0.9611	0.9611

we can observe a change in the top cluster on the first row from 14 to 11, 4 and so on. For the same method at $c = 75$ which also presented the best quality in previous experiment, we can observe that the top cluster remains equal to 6 for movements of users by as much as 1750 meters in either direction.

Table 3
Results of mobility for different amounts of user movement

		No move		$250m$		$500m$		$750m$		$1000m$		$1250m$		$1500m$		$1750m$		$2000m$	
Config	N	I	R	I	R	I	R	I	R	I	R	I	R	I	R	I	R	I	R
GK-V	1	14	4.112	11	4.091	4	4.109	11	4.045	11	4.120	11	3.782	13	4.032	11	3.849	11	3.826
Paris	2	11	4.121	13	4.157	11	4.120	13	4.124	14	4.153	14	3.915	4	4.032	14	3.857	4	4.116
$c = 15$	3	13	4.181	4	4.165	14	4.121	4	4.152	3	4.178	13	3.931	11	4.044	13	3.894	14	4.116
GK-V	1	**6**	3.320	**6**	3.320	**6**	3.320	**6**	3.344	**6**	3.428	**6**	3.419	**6**	3.311	**6**	3.424	60	3.795
Paris	2	60	4.063	34	4.036	34	3.448	34	4.065	34	3.773	34	3.771	34	3.804	34	3.781	59	3.796
$c = 75$	3	25	4.107	60	4.063	60	3.978	60	4.069	60	3.947	11	4.124	60	4.010	60	4.087	6	3.805
FCM-V	1	**1**	2.315	**1**	2.341	**1**	2.341	**1**	2.315	**1**	2.320	**1**	2.341	**1**	2.347	**1**	2.395	3	2.397
Vienna	2	**6**	2.378	**6**	2.384	**6**	2.378	**6**	2.352	7	2.369	6	2.352	6	2.381	7	2.398	7	2.436
$c = 10$	3	**7**	2.399	**7**	2.399	**7**	2.399	**7**	2.399	3	2.375	7	2.368	3	2.473	6	2.417	1	2.437
GK-V	1	**1**	2.320	**1**	2.347	**1**	2.341	**1**	2.320	**1**	2.347	3	2.346	1	2.367	10	2.385	3	2.377
Vienna	2	**3**	2.373	**3**	2.373	**3**	2.343	**3**	2.341	**3**	2.347	1	2.352	3	2.386	1	2.395	7	2.398
$c = 10$	3	**10**	2.375	**10**	2.399	**10**	2.388	**10**	2.405	**10**	2.393	7	2.372	7	2.436	3	2.395	10	2.449

For the Vienna dataset, as shown in the lower half of the Table 3, using either FCM with Voronoi or GK clustering with Voronoi tessellation keeps the top 3 clusters unchanged up to about 1000 meters of random movement.

6.5. Quality of clustering the POIs

For an extensive analysis of the effect of initial number of clusters, the POIs of both cities were clustered using FCM and GK methods with several initial numbers of clusters. The desired minimum number of clusters will depend on several parameters including the total number of POIs. In practice, there is a *lower limit* on number of clusters, since with small number of clusters each cluster will contain a large portion of POIs and will not contribute to our main goal of pruning the search space and improving the performance. There is also a *higher limit* on number of clusters for each city. Large number of clusters will have a negative effect since we should search over all cluster centers to find the best candidate clusters. This will become a time consuming process itself, hindering the desired performance improvement. Considering both limitations, we selected the range of values $10 \leqslant c \leqslant 100$ for Paris and $10 \leqslant c \leqslant 70$ for Vienna, to focus on the potentially useful intervals.

The quality indices are depicted in Fig. 4 and Fig. 5 for the Paris and Vienna datasets respectively. Recall from Section 5.2 that PC and CE indices have their ideal values at their maximum, while SC and XB have their ideal values at their minimum.

For the Paris dataset and FCM clustering, shown in Fig. 4(a), the PC index which measures the overlapping between fuzzy clusters is decreasing for $c = 15$ and above since the overlap increases with higher number of clusters. Local optimums can be seen at $c = 65$ and $c = 75$. For GK clustering, shown in Fig. 4(c), this index is similar to FCM, but increases again for $c = 70$ and above. It can be interpreted as GK clusters beginning to take better shapes than FCM clusters at this point.

The CE index is almost monotonically increasing as shown in Fig. 4(a) and (c). A small change of its behavior can be observed at about $c = 70$, which will help us to make decisions when used in combination with other indices. In Fig. 4(b) the SC index has several local optimums for FCM, but has the same monotonic behavior for GK clustering as can be seen in Fig. 4(d). The interesting index for this dataset is XB, shown in lower diagrams of Fig. 4(b) and (d). It reflects more variations than other three indices. For small number of clusters, the XB index is high. It can be interpreted as non-optimality of such cluster numbers which are near the lower limit. The local minima points for FCM are at $c = 20, 40,$

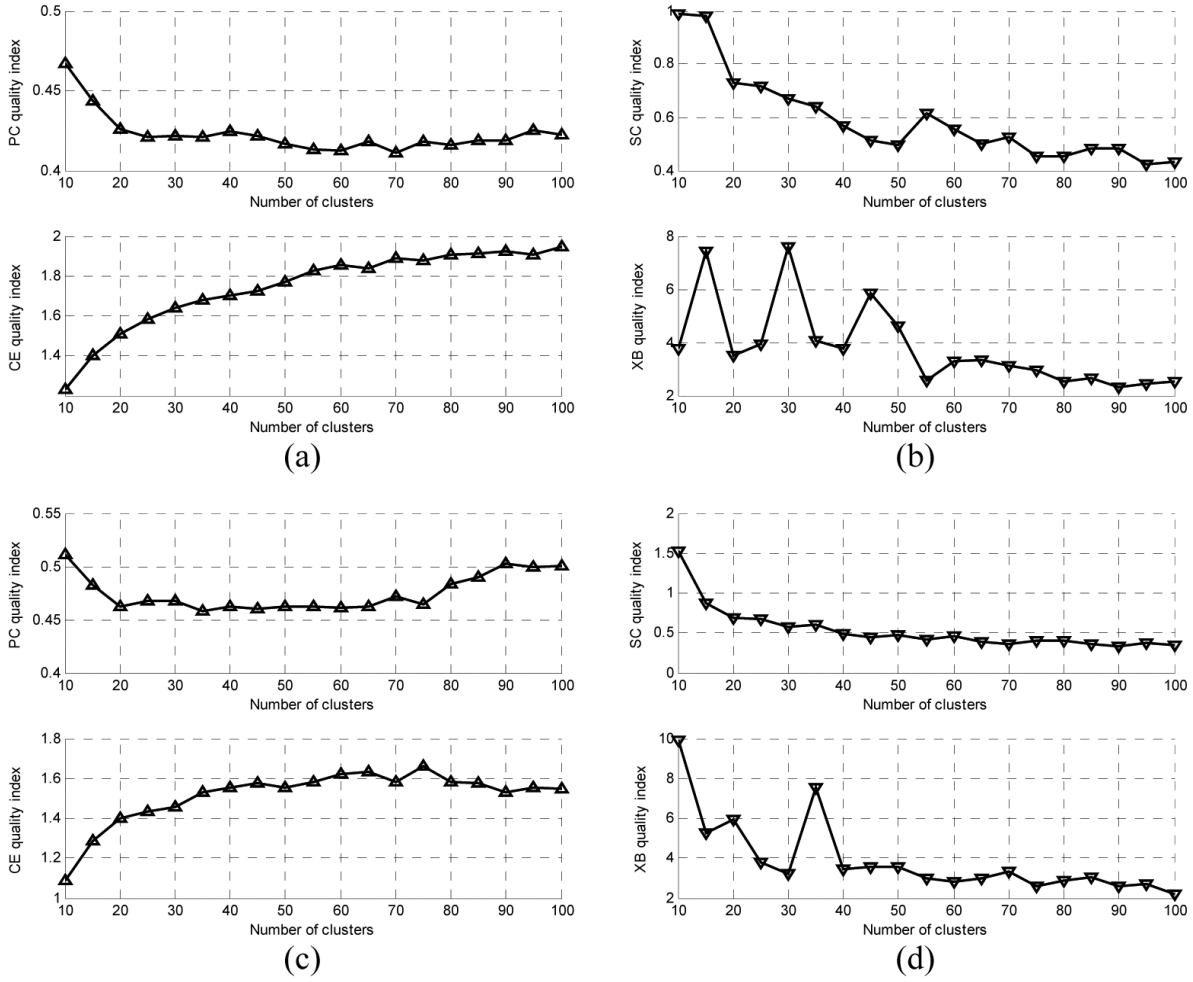

Fig. 4. The results of evaluating the quality of clustering for Paris dataset, using FCM is shown in (a), (b) and using GK clustering in (c), (d).

55, 90 and for GK at $c = 15$, 30, 40, 60, 75, 90 where we expect better quality of clustering. Therefore, we will focus on these points in next experiments to examine our proposed method of clustering and tessellation from other aspects.

Figure 5 shows the results of evaluating the quality of clustering for the Vienna dataset. The PC index in is monotonic again, but in opposite direction for both FCM shown in Fig. 5(a) and for GK clustering shown in Fig. 5(c). The increasing behavior of this index should not be interpreted as a indication of better clustering. For high number of clusters, e.g. $c > 40$ in this dataset which contains only 195 POIs, many clusters will contain only 2–3 POIs, or a single POI for outer regions. The SC index shown in Fig. 5(b) and Fig. 5(d) has the same problem with its constantly decreasing value. The CE index has a couple of interesting (local maximum) points at $c = 25$, 30 for FCM shown in Fig. 5(a) and at $c = 20$ for GK clustering shown in Fig. 5(c). The XB index in Fig. 5(b) and Fig. 5(d) is again more interesting than other three indices for Vienna dataset. We will focus on $c = 10$, 20, 40, 50 which correspond with local optimums of the last two indices either for FCM, GK or both.

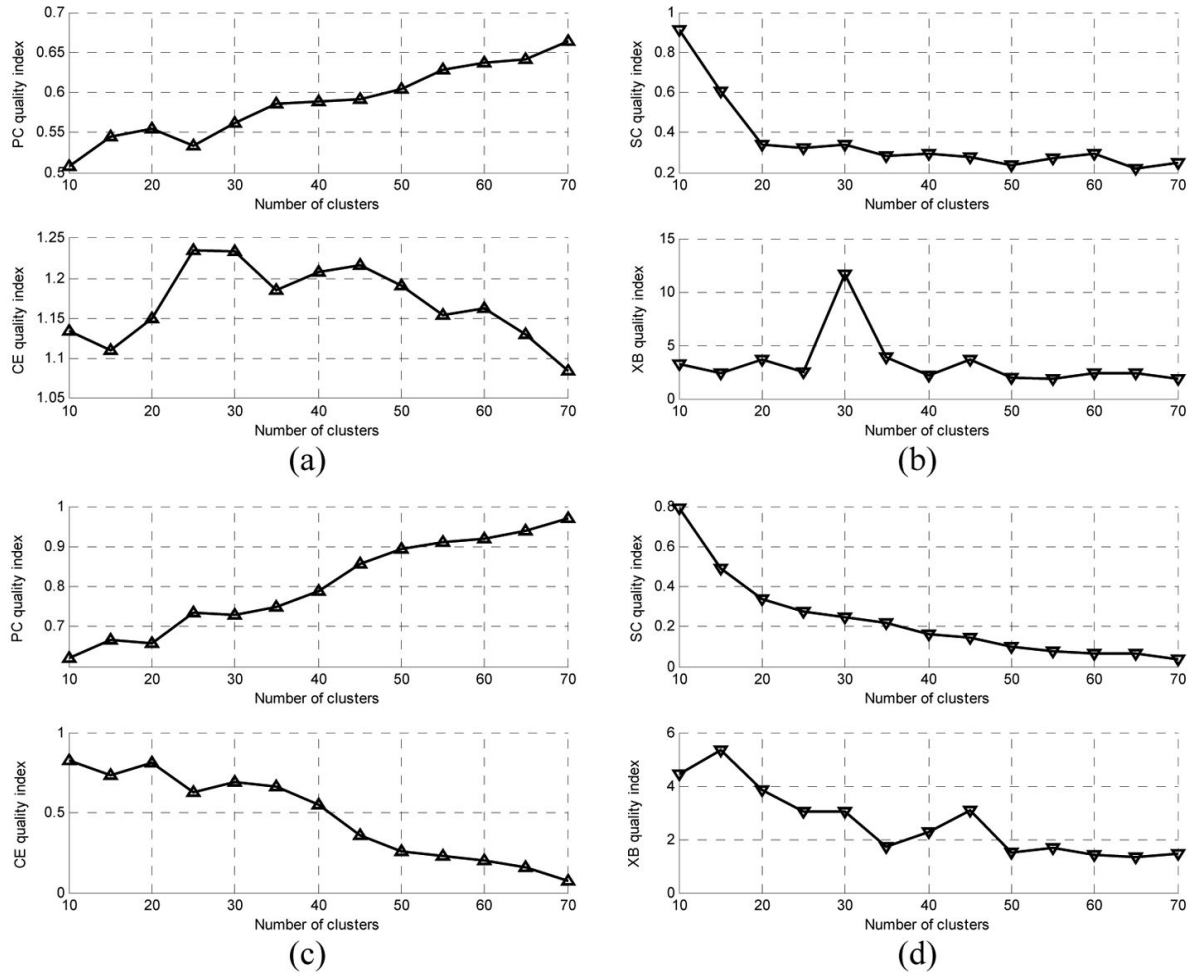

Fig. 5. The results of evaluating the quality of clustering for Vienna dataset, using FCM is shown in (a), (b) and using GK clustering in (c), (d).

It should be remembered that although XB or other measures may exhibit near-optimum values for $c > 80$, the increased number of clusters will degrade the performance of GREST-based GNN queries. For smaller number of POIs like Vienna, high number of clusters ($c > 50$ or more) leads to very slow convergence since each cluster will contain a very small number of POIs. We observed some irregular cluster shapes with unnecessary large number of clusters.

6.6. Summary of key results

Based on our experiments, we can summarize some key findings that conform to our initial requirements:

– The quality of approximation based on the type-2 fuzzy Jaccard similarity measure agrees with optimum number of clusters and shows that fuzzy clustering augmented with Voronoi tessellation is a very good approximation to original GREST-based GNN query processing method of [12].

- Clustering improves the performance up to 10 times, without losing the quality of approximation, i.e. the similarity value of 1.000 between ideal and actual top-k GREST-based GNN results.
- We observed that Voronoi diagrams improve both FCM and GK clustering; GK clustering performed better than FCM when combined with Voronoi tessellation, at the local optimum points of the XB quality index.
- The number of clusters is an important factor. In fact, a trade-off between response time and the quality of clustering is required which also affects the quality of approximation.
- In GK clustering when we have local optimum value of the XB quality index, we observed the best speed-up factor both for Paris and Vienna datasets. By augmenting the clustering at those local optimum points with Voronoi tessellation, we achieved a better quality of approximation than non-Voronoi modes.
- The user can make a decision on initial number of clusters by selecting the values corresponding to local optimum points of XB index for GK clustering. Such values minimize the run time and provide good quality of approximation.
- The effect of size of the city; for the larger city of Paris the GK clustering exhibits a great difference with FCM in quality of approximation. For a smaller city like Vienna, all methods provided good performance for $c < 40$ clusters with a very small difference in the quality of approximation.
- Robustness to the movement of people up to about 1000 meters is provided when using GK clustering combined with tessellation for the larger city, and by both methods for the smaller city.

7. Conclusion

In this paper we proposed a novel method for efficient group-based GREST-based GNN query processing. Two well-known fuzzy clustering methods were extensively investigated using several evaluation criteria, over two datasets of POIs representing a large and a medium-sized city. The process also utilized spatial tessellation which augmented the clustering by better shaping of cluster boundaries, especially the irregular regions resulted from GK clustering.

The experimental results which conformed to our initially defined requirements provided a performance improvement of up to ten times the naïve method, with a high quality of approximation as computed by the type-2 fuzzy logic similarity measure.

Future work may include other aspects of distance measures for specific application areas, in addition to the spatial, temporal and economical distances.

References

[1] R. Babuka, P.J. van der Veen and U. Kaymak, Improved covariance estimation for Gustafson-Kessel clustering, in: *Fuzzy Systems, 2002. FUZZ-IEEE'02, Proceedings of the 2002 IEEE International Conference on*, 2002, pp. 1081–1085.
[2] B. Balasko, J. Abonyi and B. Feil, *Fuzzy Clustering and Data Analysis Toolbox Manual*, 2005.
[3] A.M. Bensaid, L.O. Hall, J.C. Bezdek, L.P. Clarke, M.L. Silbiger, J.A. Arrington and R.F. Murtagh, Validity-guided (re)clustering with applications to image segmentation, *Fuzzy Systems, IEEE Transactions on* **4** (1996), 112–123.
[4] S. Bereg, M. Gavrilova and Y. Zhang, Robust Point-Location in Generalized Voronoi Diagrams, in, 2009, pp. 285–299.
[5] J. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Kluwer Academic Publishers Norwell, MA, USA, 1981.
[6] S. Caballé, F. Xhafa and L. Barolli, Using mobile devices to support online collaborative learning, *Mobile Information Systems* **6** (2010), 27–47.
[7] M.A. de Leite and I.L.M. Ricarte, Fuzzy Information Retrieval Model Based on Multiple Related Ontologies, in: *Tools with Artificial Intelligence, 2008. ICTAI '08. 20th IEEE International Conference on*, 2008, pp. 309–316.

[8] K. Deng, H. Xu, S. Sadiq, Y. Lu, G. Fung and H.T. Shen, Processing Group Nearest Group Query, in: *Data Engineering, 2009. ICDE '09. IEEE 25th International Conference on*, 2009, pp. 1144–1147.
[9] M.M. Deza and E. Deza, Encyclopedia of Distances, in: *Encyclopedia of Distances*, 2009, pp. 1–583.
[10] A. Durresi and M. Denko, Advances in mobile communications and computing, *Mobile Information Systems* **5** (2009), 101–103.
[11] N. Ghadiri, A. Baraani, N. Ghasem-Aghaee and M. Nematbakhsh, A Human-Centric Approach To Group-Based Context-Awareness, *International Journal of Network Security and its Applications* **3** (2011), 47–66.
[12] N. Ghadiri, A. Baraani, N. Ghasem-Aghaee and M. Nematbakhsh, GREST – A Type-2 Fuzzy Distance Model for Group Nearest-Neighbor Queries, *Submitted* (2010).
[13] S.R. Gulliver, G. Ghinea, M. Patel and T. Serif, A context-aware Tour Guide: User implications, *Mobile Information Systems* **3** (2007), 71–88.
[14] D.E. Gustafson and W.C. Kessel, Fuzzy clustering with a fuzzy covariance matrix, in: *Decision and Control including the 17th Symposium on Adaptive Processes, 1978 IEEE Conference on*, 1978, pp. 761–766.
[15] A.M. Hanashi, I. Awan and M. Woodward, Performance evaluation with different mobility models for dynamic probabilistic flooding in MANETs, *Mobile Information Systems* **5** (2009), 65–80.
[16] T. Hashem, L. Kulik and R. Zhang, Privacy preserving group nearest neighbor queries, in: *Proceedings of the 13th International Conference on Extending Database Technology*, ACM, Lausanne, Switzerland, 2010, pp. 489–500.
[17] F. Höppner and F. Klawonn, Improved fuzzy partitions for fuzzy regression models, *International Journal of Approximate Reasoning* **32** (2003), 85–102.
[18] H.-H. Hsu and C.-C. Chen, RFID-based human behavior modeling and anomaly detection for elderly care, *Mobile Information Systems* **6** (2010), 341–354.
[19] X. Hu, L. Yansheng and L. Zhicheng, Continuous Group Nearest Group Query on Moving Objects, in: *Education Technology and Computer Science (ETCS), 2010 Second International Workshop on*, 2010, pp. 350–353.
[20] J. Jayaputera and D. Taniar, Data retrieval for location-dependent queries in a multi-cell wireless environment, *Mobile Information Systems* **1** (2005), 91–108.
[21] T. Kwok, K. Smith, S. Lozano and D. Taniar, Parallel Fuzzy c- Means Clustering for Large Data Sets, in: *Euro-Par 2002 Parallel Processing*, B. Monien and R. Feldmann, eds, Springer Berlin / Heidelberg, 2002, pp. 27–58.
[22] H. Li, H. Lu, B. Huang and Z. Huang, Two ellipse-based pruning methods for group nearest neighbor queries, in: *Proceedings of the 13th annual ACM international workshop on Geographic information systems*, ACM, Bremen, Germany, 2005.
[23] X. Lian and L. Chen, Probabilistic Group Nearest Neighbor Queries in Uncertain Databases, *Knowledge and Data Engineering, IEEE Transactions on* **20** (2008), 809–824.
[24] F. Liu and J.M. Mendel, Encoding Words into Interval Type-2 Fuzzy Sets Using an Interval Approach, *Fuzzy Systems, IEEE Transactions on* **15** (2008).
[25] Z. Mammeri, F. Morvan, A. Hameurlain and N. Marsit, Location-dependent query processing under soft real-time constraints, *Mobile Information Systems* **5** (2009), 205–232.
[26] J.M. Mendel, Computing with words and its relationships with fuzzistics, *Inf Sci* **177** (2007), 988–1006.
[27] J.M. Mendel, Type-2 Fuzzy Sets and Systems: An Overview [corrected reprint], *Computational Intelligence Magazine, IEEE* **2** (2007), 20–29.
[28] J.M. Mendel, On answering the question "Where do I start in order to solve a new problem involving interval type-2 fuzzy sets?", *Information Sciences* **179** (2009), 3418–3431.
[29] J.M. Mendel, R.I. John and F. Liu, Interval Type-2 Fuzzy Logic Systems Made Simple, *Fuzzy Systems, IEEE Transactions on* **14** (2006), 808–821.
[30] D. Papadias, Q. Shen, Y. Tao and K. Mouratidis, Group nearest neighbor queries, in: *Data Engineering, 2004. Proceedings. 20th International Conference on*, 2004, pp. 301–312.
[31] D. Papadias, Y. Tao, K. Mouratidis and C.K. Hui, Aggregate nearest neighbor queries in spatial databases, *ACM Trans Database Syst* **30** (2005), 529–576.
[32] W. Pedrycz, A. Skowron and V. Kreinovich, *Handbook of Granular Computing*, Wiley-Interscience, 2008.
[33] I. Priggouris, D. Spanoudakis, M. Spanoudakis and S. Hadjiefthymiades, A generic framework for Location-Based Services (LBS) provisioning, *Mobile Information Systems* **2** (2006), 111–133.
[34] S. Rovetta and F. Masulli, Vector quantization and fuzzy ranks for image reconstruction, *Image and Vision Computing* **25** (2007), 204–213.
[35] M. Safar, Group K -Nearest Neighbors queries in spatial network databases, *Journal of Geographical Systems* **10** (2008), 407–416.
[36] M. Safar, D. Ibrahimi and D. Taniar, Voronoi-based reverse nearest neighbor query processing on spatial networks, *Multimedia Systems* **15** (2009), 295–308.
[37] M. Sharifzadeh and C. Shahabi, VoR-Tree: R-trees with Voronoi Diagrams for Efficient Processing of Spatial Nearest Neighbor Queries, *Proceedings of the VLDB Endowment* **3** (2010).

[38] G. Trajcevski, A. Choudhary, O. Wolfson, L. Ye and G. Li, Uncertain Range Queries for Necklaces, in: *Mobile Data Management* (*MDM*), *2010 Eleventh International Conference on*, 2010, pp. 199–208.
[39] A. Waluyo, B. Srinivasan and D. Taniar, Research on location-dependent queries in mobile databases, *International Journal of Computer Systems Science & Engineering* **20** (2005), 79–95.
[40] D. Wu and J.M. Mendel, Aggregation Using the Linguistic Weighted Average and Interval Type-2 Fuzzy Sets, *Fuzzy Systems, IEEE Transactions on* **15** (2007), 1145–1161.
[41] D. Wu and J.M. Mendel, A comparative study of ranking methods, similarity measures and uncertainty measures for interval type-2 fuzzy sets, *Information Sciences* **179** (2009), 1169–1192.
[42] X.L. Xie and G. Beni, A validity measure for fuzzy clustering, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **13** (1991), 841–847.
[43] K. Xuan, G. Zhao, D. Taniar, W. Rahayu, M. Safar and B. Srinivasan, Voronoi-based range and continuous range query processing in mobile databases, *Journal of Computer and System Sciences* **In Press, Corrected Proof** (2010).
[44] K. Xuan, G. Zhao, D. Taniar, M. Safar and B. Srinivasan, Voronoi-based multi-level range search in mobile navigation, *Multimedia Tools and Applications* (2010), 1–21.
[45] K. Xuan, G. Zhao, D. Taniar and B. Srinivasan, Continuous Range Search Query Processing in Mobile Navigation, in: *Parallel and Distributed Systems, 2008. ICPADS '08. 14th IEEE International Conference on*, 2008, pp. 361–368.
[46] J. Yang and Y. Ning, Research on feature weights of fuzzy c-means algorithm and its application to intrusion detection, in: *Environmental Science and Information Application Technology* (*ESIAT*), *2010 International Conference on*, 2010, pp. 164–166.
[47] M.L. Yiu, N. Mamoulis and D. Papadias, Aggregate nearest neighbor queries in road networks, *Knowledge and Data Engineering, IEEE Transactions on* **17** (2005), 820–833.
[48] L.A. Zadeh, From computing with numbers to computing with words. From manipulation of measurements to manipulation of perceptions, *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on* **46** (1999), 105–119.
[49] G. Zhao, K. Xuan, W. Rahayu, D. Taniar, M. Safar, M. Gavrilova and B. Srinivasan, Voronoi-Based Continuous k Nearest Neighbor Search in Mobile Navigation, *Industrial Electronics, IEEE Transactions on* (2010).
[50] G. Zhao, K. Xuan, D. Taniar, M. Safar, M. Gavrilova and B. Srinivasan, Multiple Object Types KNN Search Using Network Voronoi Diagram, in: *Computational Science and Its Applications – ICCSA 2009*, O. Gervasi, D. Taniar, B. Murgante, A. Laganà, Y. Mun and M. Gavrilova, eds, Springer Berlin / Heidelberg, 2009, pp. 819–834.

Nasser Ghadiri is a PhD candidate of computer engineering at the Faculty of Engineering of the University of Isfahan (UI). He earned his M.Sc and B.Sc degrees from the University of Shiraz and Isfahan University of Technology, respectively. His research interests are spatial and mobile databases, computational intelligence and service-oriented architectures. He is a member of IEEE and ACM.

Ahmad Baraani-Dastjerdi is an assistant professor of computer engineering at the School of Engineering of the University of Isfahan (UI). He got his BS in Statistics and Computing in 1977. He got his MS & PhD degrees in Computer Science from George Washington University in 1979 & University of Wollongong in 1996, respectively. He is Head of the Research Department of the Communication systems and Information Security (CSIS) and Head of the ACM International Collegiate Programming Contest (ACM/ICPC) of University of Isfahan from 2000 until present. He co-authored three books in Persian and received an award of "the Best e-Commerce Iranian Journal Paper" (2005). Currently, he is teaching PhD and MS courses of Advance Topics in Database, Data Protection, Advance Databases, and Machining Learning. His research interests lie in Databases, Data security, Information Systems, e-Society, e-Learning, e-Commerce, Security in e-Commerce, and Security in e-Learning.

Nasser Ghasem-Aghaee is a professor of computer engineering at the Faculty of Engineering of the University of Isfahan (UI) and Sheikh-Bahaei University. He earned his PhD & MSc degrees from the University of Bradford and Georgia Tech, respectively. He spent two sabbatical leave (1993–94 & 2002–03) at the Ottawa Center of the McLeod Institute of Simulation Sciences, at Computer Science Department of the University of Ottawa, Ottawa, Ontario, Canada. He served as his Department Chair and Research and Graduate Studies Deputy Manager of Engineering College at the University of Isfahan between 1987 and 1993 and From 1994 until now, respectively. He authored three books in Persian and published more than 70 documents. He has been active in seminars and conferences held in different countries. His research interests have been in areas of Computer Simulation, Object-Oriented Analysis and Design, Artifical Intelligence (AI) and Expert Systems, AI in Software Engineering, AI in Simulation, OO in Simulation, AI in Object-Oriented Analysis, User Modeling, Advance Artificial Intelligence, and Software Agents and Applications.

Mohammad A. Nematbakhsh is an associate professor of computer engineering at the School of Engineering of the University of Isfahan (UI). He received his BSc in Electrical Engineering from Louisiana Tech University, USA, in 1981 and his MSc & PhD degrees in Electrical and Computer Engineering from University of Arizona, USA, in 1983 & 1987, respectively. He has published more than 70 papers and 3 US patents, and authored a book on database systems that is widely used in universities. He has received five awards and was the chair of the 6th CSI Computer Engineering Conference in 2001. He has been distinguished research fellow at the University of Isfahan and he was also awarded as the best national thesis advisor. He is the member of editorial board of several journals in Engineering Sciences. His main research interests include multi-agent systems applications in e-commerce and computer networks.