Multimed Tools Appl (2011) 53:459–479
DOI 10.1007/s11042-010-0498-y

Voronoi-based multi-level range search in mobile navigation

Kefeng Xuan · Geng Zhao · David Taniar · Maytham Safar · Bala Srinivasan

Published online: 26 March 2010

Abstract Due to the universality and importance of range search queries processing in mobile and spatial databases as well as in geographic information system (GIS), numerous approaches on range search algorithms have been proposed in recent years. But ordinary range search queries focus only on a specific type of point objects. For queries which require to retrieve objects of interest locating in a particular region, ordinary range search could not get the expected results. In addition, most existing range search methods need to perform a searching on each road segments within the pre-defined range, which decreases the performance of range search. In this paper, we design a weighted network Voronoi diagram and propose a high-performance multilevel range search query processing that retrieves a set of objects locating in some specified region within the searching range. The experimental results show that our proposed algorithm runs very efficiently and outperforms its main competitor.

Keywords Mobile databases · Voronoi diagram · Range queries · Mobile navigation

K. Xuan (✉) · G. Zhao · D. Taniar · B. Srinivasan
Clayton School of Information Technology, Monash University, Melbourne, Australia
e-mail: Kefeng.Xuan@infotech.monash.edu.au

G. Zhao
e-mail: Geng.Zhao@infotech.monash.edu.au

D. Taniar
e-mail: David.Taniar@infotech.monash.edu.au

B. Srinivasan
e-mail: Bala.Srinivasan@infotech.monash.edu.au

M. Safar
Computer Engineering Department, Kuwait University, Kuwait City, Kuwait
e-mail: Maytham.Safar@ku.edu.kw

1 Introduction

Range searching has been extensively discussed in many areas including data structures, information retrieval, computational geometry, wireless communication [15] spatial and mobile databases [26, 27] for years. The research results are also implemented in the industrial fields, such as search engine, geographic information system (GIS), global positioning system (GPS) and digital map. A qualified application (Web applications: Google Maps, Bing Maps and Mobile devices: PDAs, cellular phones, car navigation systems) for range searching must be competent for a variety of queries in any complex environment and can respond queries accurately and efficiently. Figure 1 illustrates an example of range search in a digital map. The user (query point q) wants to get all petrol stations within 3kms from current location of user or the query location. In Fig. 1, the object of interest (petrol station) is represented by a sequence of numbers. The distance from query point to each object is the shortest road network connection rather than the straight line. Then the objects within 3kms to query point are highlighted with red while the green objects indicate the petrol stations are out of 3kms to the query point. This simple question has been answered by many existing work, through the use of spatial indexing [3–5, 11, 22, 23] and various spatial query algorithms [17, 30].

But in reality, range search query not only involves traveling distance, but also requires the objects of interest belong to a particular region (such as, university campus, suburb, shopping mall). For example, the user would like to find car parks locating in a university campus within 1.5kms. See Fig. 2, the car parks from query point within 1.5kms are 1, 2, 3, 4, 5, but only 1, 2, 3 are located in the pre-defined region (university campus). So object 4 and 5 have to be removed from the result list before it is returned to the user. Intuitively, the objective of this type of range search query includes two steps. The first step is finding the pre-defined region within the

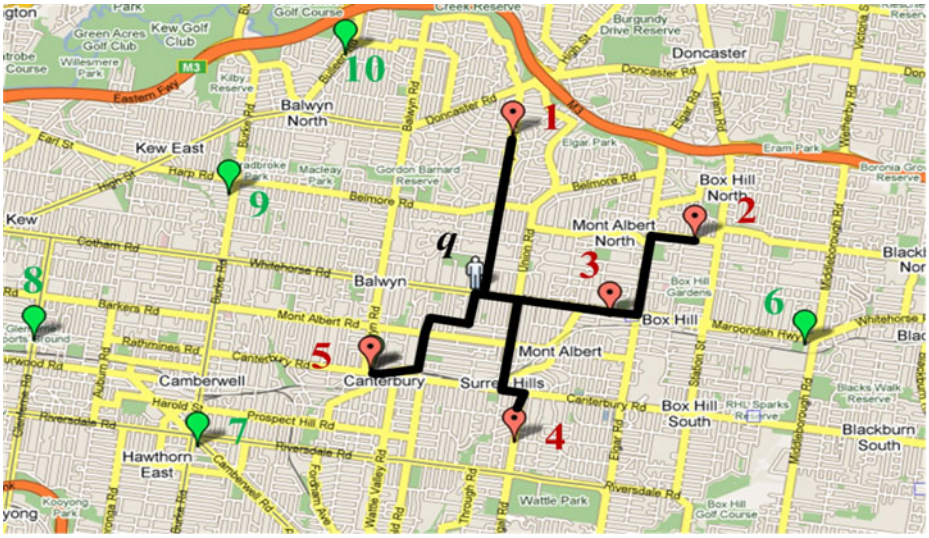


Fig. 1 An example of range search

Fig. 2 An example of region constrained range search

searching range—the object of interest (4) that is even closer to query point but is not in the pre-defined region will not be considered. The second step is estimating each object of interest in the pre-defined region to check whether it is within the searching range or not. The ordinary range search approaches are not concerned about objects of interest in a large region, and obviously, these approaches are incapable to solve region constrained range queries.

In order to address this problem, we proposed *Region Constrained Range* (RCR) in our previous work. RCR requires to construct several network Voronoi diagrams (NVD), which needs a mass of storage space and pre-compaction. The performance of RCR decreases dramatically when the searching range increases greatly. In this paper, we propose a better solution for region constrained range search, called *Multi-level Range Search* (MRS). Our experimental results show that *MRS* outperforms *RCR* in most scenarios for mobile navigation.

2 Voronoi diagram in mobile navigation search: background

Voronoi diagram (VD) [16] and network Voronoi diagram (NVD) are a special type of decomposition for a metric space determined by a set of discrete objects. Their geometric properties have been widely used to process mobile and spatial queries. Thereafter, a special VD, weighted Voronoi diagram (WVD) has been extensively studied in [2]. In this paper, we propose network weighted Voronoi diagram that can be used to process spatial queries on a set of regions or large objects as a variation of WVD.

2.1 VD and NVD

Voronoi Diagram partition the Euclidean plane into a set of convex polygons, named *Voronoi polygon* or *Voronoi region*. Each Voronoi polygon involves a generator

(point) to which the Euclidean distance d_e from any point in its polygon is smaller than to any other generator. Voronoi polygon is composed by several Voronoi edges, also called borders, which is always shared by a pair of adjacent polygons. Any point on the Voronoi edge has the same distance to the pair of generators that associate with this edge. The formal definition of Voronoi Diagram is given as follow:

Definition 1 Given a set of discrete objects $\wp = (P_1, P_2, ..., P_n)$, $n \in$ Integer I_n, in Cartesian space, the Voronoi polygon of P_i, $VP(P_i)$, is a universal of points that satisfy:

$$\{\forall p | d_e(p, P_i) \leq d_e(p, P_j)\}, (i, j \in I_n, i \neq j, P_i, P_j \in \wp)$$

The Voronoi diagram for $\wp$ is:

$$VD(\wp) = \bigcup_{i=1}^{n} VP(P_i)$$

The function $d_e(p, P_i)$ denotes the distance from point p to P_i, which can be calculated by:

$$d_e(p, P_i) = \sqrt{\left(x_p^2 - x_{P_i}^2\right) + \left(y_p^2 - y_{P_i}^2\right)}$$

x, y are the aces in a coordinate system. In term of the definition of VP, for any point p inside $VP(P_i)$, the distance from it to P_i must be the minimum one comparing to the distances to other objects. The equality in the definition of Voronoi polygon holds for the points on the borders of VP_{P_i} and VP_{P_j}.

Figure 3 shows an example of VD, in which, any point s in VP_{P_2}, whose distance to generators P_1, P_2 are $d_e(s, P_1)$ and $d_e(s, P_2)$ satisfying $d_e(s, P_1) < d_e(s, P_2)$. Since point b is on the shared edge E of VP_{P_1} and VP_{P_2}, then $d_e(b, P_1) = d_e(b, P_2)$.

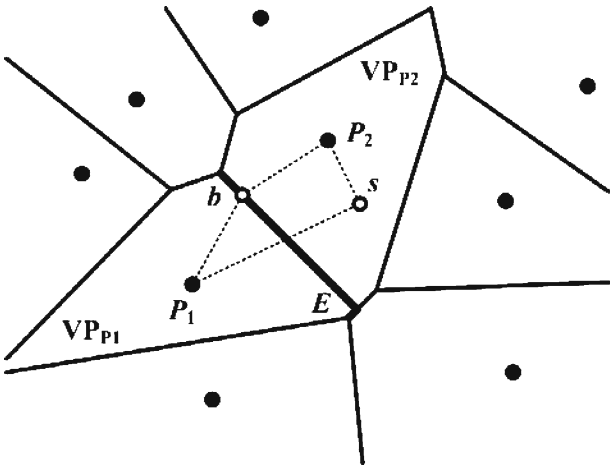


Fig. 3 Voronoi Diagram (VD)

Network Voronoi Diagram (NVD) is a special Voronoi diagram constructed on road networks involving a set of nodes and links rather than Euclidean plane. In NVD, the partitions are a set of road segments termed network Voronoi polygon (NVP) instead of Voronoi polygon. For each NVP, there is only one generator. Any point p in $NVP(P_i)$ also satisfies:

$$\{\forall p | d_{net}(p, P_i) \leq d_{net}(p, P_j)\}, (i, j \in I_n, i \neq j, P_i, P_j \in \wp)$$

The edges of Voronoi polygons shrink to the midpoints of the road network connection between two objects of interest. Beside the object of interest, the intersections (white point) of the networks are also presented in the NVD. Referring to Fig. 4, the objects of interest (dark points) act as the generators of NVPs that are distinguished by different line styles.

With NVD, a spatial network can be modeled as a weighted graph. All road network intersections and objects can be represented as nodes in the weighted graph. The link connecting these nodes represents road networks while the distances are the weight for these links.

The distance which is the shortest route among road network connections from one node to another can be calculated by using Dijkstra's algorithm [6] or other expansion techniques [17]. When the distances from each border point to object of interest/other border points are calculated off-line (NVD stores the relative position of the objects in the road networks), the approaches based on NVD can load more required information from spatial/mobile database every time, which improves the performance of spatial and mobile queries processing.

2.2 Weighted Voronoi diagram

In ordinary Voronoi diagrams all generators are identical and each generator has the same weight, the Voronoi edges are straight and the Voronoi polygons are contiguous. The weighted Voronoi diagram (multiplicatively, additively, compound) differs from ordinary Voronoi diagram in that the generators have different weight

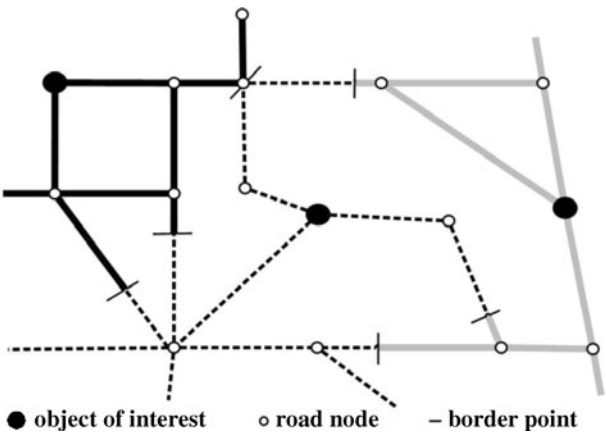


Fig. 4 Network Voronoi Diagram (NVD)

which denotes as ω [16]. In this paper, we focus on additively weighted (AW) Voronoi diagram only. The definition of AW Voronoi diagram is as follow:

Definition 2 Given a set of discrete objects $\wp = (P_1, P_2,..., P_n)$, $n \in$ Integer I_n, in Cartesian space. Each point P is assigned a weight ω_P. the weighted Voronoi polygon of $P_i \in \wp$, $VP(P_i)$, includes all points that satisfy:

$$\{\forall p | d_e(p, P_i) - \omega_{P_i} \leq d_e(p, P_j) - \omega_{P_j}\}, (i, j \in I_n, i \neq j, P_i, P_j \in \wp)$$

The additively weighted Voronoi diagram for $\wp$ is:

$$WVD(\wp) = \bigcup_{i=1}^{n} WVP(P_i)$$

In the two-dimension space, AW-Voronoi diagram can be seen as a standard Voronoi diagram of a set of rotundities each centered at a point P with ω_P as a radius. According to the definition of weighted Voronoi diagram, each weighted Voronoi polygon is assigned to a unique rotundity which is the closest area of all points inside that polygon. The points p on borders of weighted Voronoi polygons P_i and P_j satisfy:

$$\{\forall p | d_e(p, P_i) - \omega_{P_i} = d_e(p, P_j) - \omega_{P_j}\}$$

Figure 5 illustrates an example of AW-Voronoi diagram of five weighted objects (R) centered at Voronoi generators (P) with different weights (ω_P). It is obvious that for any point s in weighted Voronoi polygon of R_1 has:

$$d_e(s, R_1) \leq d_e(s, R_2)$$
$$d_e(s, R_1) = d_e(s, P_1) - \omega_{P_1}$$
$$d_e(s, R_2) = d_e(s, P_2) - \omega_{P_2}$$

and for any point b on the border of R_1 and R_2 has:

$$d_e(b, R_1) = d_e(b, R_2)$$
$$d_e(b, R_1) = d_e(b, P_1) - \omega_{P_1}$$
$$d_e(b, R_2) = d_e(s, P_2) - \omega_{P_2}$$

2.3 Weighted network Voronoi diagram

In this paper we designed a novel Voronoi diagram, named weighted network Voronoi diagram (WNVD), as the underlying framework for our proposed range search algorithm. Weighted network Voronoi diagram differs from weighted Voronoi diagram and ordinary network Voronoi diagram. Weighted objects cannot be expressed as points when generating network Voronoi diagrams, as their weights should not be ignored. Normally, a weighted object occupies a big area, such as, national parks, shopping centers and school campuses. Since the path from a random point on road networks to a weighted object have to pass through one of a set of entry or exit points (we named this kind of points as access points and denoted as *ap*) of the weighted object, the distance to a weighted object is expressed by the

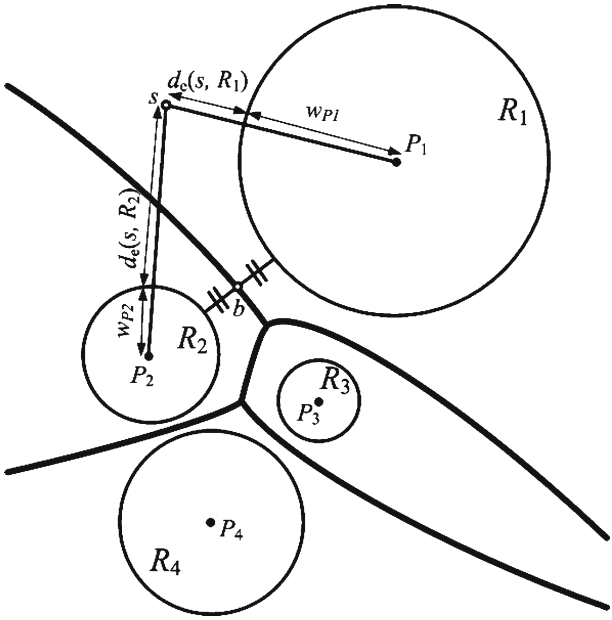


Fig. 5 Additively weighted Voronoi diagram

shortest path from point p to one of its access points, denoted as $Min(d_{net}(p, AP))$, $AP = (ap_1, ap_2, ..., ap_n)$. The formal definition of an access point is:

Definition 3 Access points AP_k are the set of intersections between edges Ed of weighted object R_k and road networks RN.

$$AP_k = \bigcup_{m=1}^{n} ap_{k\_m} = Ed(R_k) \cap RN$$

According to the definition of access point, though a weighted object can have multiple access point ap, an ap can only belong to a particular weighted object. The two subscript k, m of ap indicate which weighted object it belongs to and the indexing number of this access point respectively (Fig. 6).

Then weighted Network Voronoi diagram is defined as follow:

Definition 4 Given a set of weighted objects $\Re = \{R_1, R_2, ..., R_n\}$ and road networks RN. The NVP(R_k) is a set of segments L in the road networks, $L \in RN$. Then $\forall p \in L$, have $Min(d_{net}(p, AP_k)) \leq Min(d_{net}(p, AP_l))$, $k, l \in \aleph$, $k \neq l$, $AP_k = Ed(R_k) \cap RN$, $AP_l = Ed(R_l) \cap RN$.

$$WNVD = \bigcup_{k=1}^{n} NVP(R_k)$$

In term of definition of WNVD, all details inside of weighted objects are ignored. As a weighted object can have more than one access point, the distance from a random point p to the weighted object can have multiple values, in which

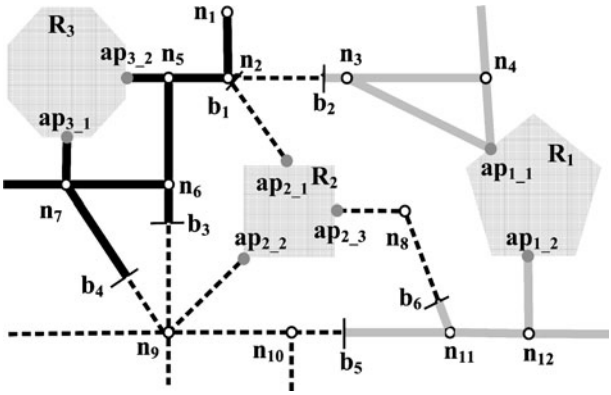


Fig. 6 Weighted Network Voronoi Diagram (WNVD)

$Min(dis(p, AP))$ and $Max(dis(p, AP))$ are two most important values indicating the smallest and longest values among $d_{net}(p, ap_1)$, $dis(p, ap_2)$, ..., $d_{net}(p, ap_n)$. $d_{net}(p, ap)$ is the length of the shortest path from p to ap, which can be calculated by Dijkstra's algorithm that can find the shortest distance from point to point off-line [6]. $Min(d_{net}(p, AP))$ is used to estimate whether the weighted object is touch the range or not, while $Max(dis(p, AP)) < e$ (e is the searching range) guarantees the weighted object is fully within the range.

For example, there are three weighted objects, namely, R_1, R_2 and R_3 each including several access points in Fig. 7. After performing Dijkstra's algorithm, the distances from point p to these weighted objects are in Table 1. If the searching range is 6kms, for R_1 the $Min(dis(p, AP)) = d_{net}(p, ap_{1_1}) = 10 > 6$, then we can say R_1 is out of range; for R_2 the $Min(dis(p, AP)) = d_{net}(p, ap_{2_2}) = 6$ and $Max(dis(p, AP)) = d_{net}(p, ap_{2_1}) = 9 > 6$, then R_2 is partially in the range; for R_3 $Min(dis(p, AP)) = Max(dis(p, AP)) = d_{net}(p, ap_{3_2}) = d_{net}(p, ap_{3_1}) = 6$, then we say R_3 is fully in the range.

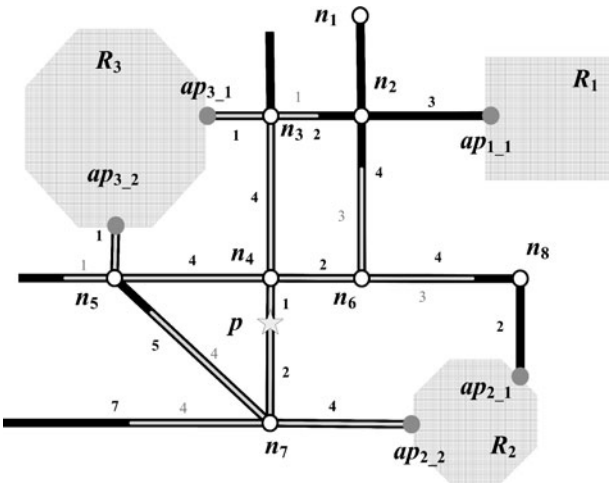


Fig. 7 Distance calculation in road networks

Table 1 Distances from p to weighted objects

To R_1 through ap_{1_1}:
$d_{net}(p, ap_{1_1}) = pn_4 + n_4n_6(n_4n_3) + n_6n_2(n_3n_2) + n_2ap_{1_1} = 10$
To R_2 through ap_{2_1}, ap_{2_2}:
$d_{net}(p, ap_{2_1}) = pn_4 + n_4n_6 + n_6n_8 + n_8ap_{2_1} = 9$
$d_{net}(p, ap_{2_2}) = pn_7 + n_4ap_{2_2} = 6$
To R_3 through ap_{3_1}, ap_{3_2}:
$d_{net}(p, ap_{3_2}) = pn_4 + n_4n_3 + n_3ap_{3_1} = 6$
$d_{net}(p, ap_{3_1}) = pn_4 + n_4n_5 + n_4ap_{3_2} = 6$

3 Multi-level range search processing

In this section, we introduce the basic Voronoi-based range search method, and then demonstrate our *Multi-level Range Search* (MRS) approach to process region constrained range search queries by explaining the details of the algorithm of MRS as well as a case study.

3.1 Preliminary: Voronoi-based range search (VRS)

Our previous work, *VRS*, is a traditional range search approach based on network Voronoi diagram [30] to retrieve all objects of interest within the range in road networks. The general process of the *VRS* can be summarized into three steps:

- Step 1: *Locating the Query Point* (*LQP*): first involves the contain(q) function, a common spatial index structure (e.g. R-Tree) to find the NVP(q) containing the query point q, and put its generator P_q into $Q_{pre}()$ that is a sorted candidates queue used to store the object of interests whose validity have been checked. $Q_{pre}()$ does not only involve the valid objects which in the expected searching range, but also a few of false hits just out of the searching range. Then a network expansion methods will be applied in the NVP(q) to get the $dis(q, P_q)$ and $dis(q, Borders)$
- Step 2: *Current Searching Range Expansion* (*CSRE*): is a NVP expansion method that adds all neighbors of valid objects which are in $Q_{pre}()$ into $Q_{pre}()$, which can expedite the searching range expansion, until the size of current searching range is comparable with the expected searching range and simultaneously, all met objects will be inserted into $Q_{pre}()$.
- Step 3: *Validating Objects* (*VO*) and *Gradually Shrinking* (*GS*): compares the searching range e and dis_{max} in $Q_{pre}()$ (VO), if $e < dis_{max}$, set the value of inside/outside property of P_{max} to false and set dis_{max} to the next largest value in $Q_{pre}()$(GS); if $e > dis_{max}$, do step 2 until $e < dis_{max}$

When the algorithm of *VRS* terminates, the format of $Q_{pre}()$ has to be in the following sequence:

$$Q_{pre}() = (..., (P_x, ..., False), (P_y, ..., True), ...)$$

The object with a false value indicates the object is out of the range (e.g P_x), while the one with a true property is the expected object in the range (e,g P_y). The algorithm of VRS is listed below:

Algorithm 1 VRS(q, e)

Input: query point:q, searching range:e
Output: $Q_{pre}()$

```
1: Contain(q) finds the generator P_q
2: Q_pre() ← Q_pre() ∪ P_q
3: if e < dis_net(q, Pq) then
4:    return P with true property in Q_pre()
5: else
6:    NVD_Expansion()
7: end if
8: return P with true property in Q_pre()
```

The NVD_Expansion() function includes step 2 and step 3 to expanding the searching range and verify the retrieved objects in Q_{pre} until all objects of interest in the searching range e are obtained.

Since MRS is based on WNVD and deals with region constrained range search (RCRS) in road networks, an NVD-based range search method is preferred as our foundation. VRS is the only NVD-based range search approach that provides a better solution than other network range search approaches by reducing the expansion area and declining the rate of false hits dramatically, so VRS is utilized as the foundation of our proposed approach.

3.2 Algorithm of multi-level range search

The MRS method invokes VRS on WNVD to get a set of weighted objects within the range and store them in Q_{wo}. At the same time, MRS also performs VRS on NVD to retrieve a set of objects of interest within the range and store them in Q_{oi}. Since the searching on different NVDs does not effect each other, VRS can be carried out synchronously on both $WNVD$ and NVD to improve the performance. Then for each valid weighted objects in Q_{wo}, we use spatial index structure (such as, R-tree) on Q_{oi} to filter the invalid objects and store the final result in $PQ()$. Figure 8 illustrates the procedure of MRS.

– WNVD/NVD Off-line Construction and On-line Network Expansion

Preparatory work of MRS is constructing a WNVD for weighted objects as well as an NVD for objects of interest. Some necessary data need to be calculated off-line by using Dijkstra's algorithm and stored into databases during this procedure, including distances from borders to access points (in WNVD), the distances from border to generators (in NVD), the distances from borders to borders (in both WNVD and NVD). Then if the query point position is detected by Contain(q) function, the distance from query point to the border points of WNVP(q) and NVP(q) can be calculated by performing on-line network expansion [17] in WNVD and NVD synchronously. The expansion will terminated if a border point is found. The on-line expansion method also computes the distance from query point to the nearest access

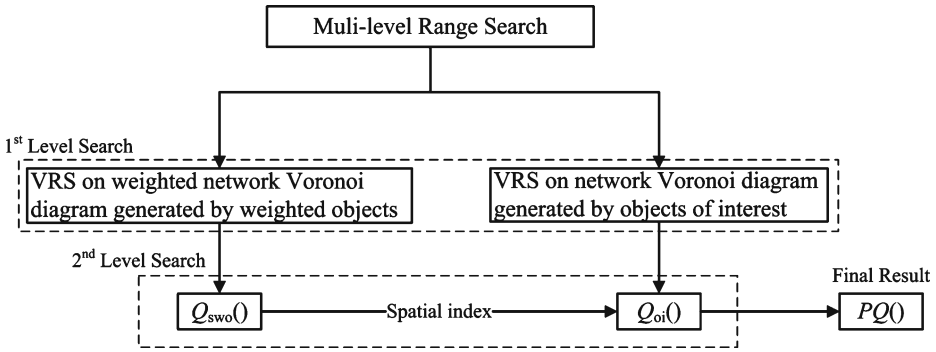


Fig. 8 Multi-level range search

point of WNVP(q) and the distance from query point to the generator of NVP(q). With off-line calculation and on-line network expansion, we minimize the disc access and improve the operational efficiency.

– Multi-level Searching

If the searching range is larger than WNVP(q) and NVP(q), we need to involve NVP expansion to get other weighted objects and objects of interest within the range. For the similarity of WNVD and NVD, we can simply perform VRS on WNVD and NVD to get Q_{wo} stored the weighted objects intersecting with searching range and Q_{oi} the objects of interest within the range, which can be seen as the first level searching. In this step, some of interest objects that do not locate in the specified weighted objects are retrieved. Hence in the next level searching, we will remove these false hits from the candidates list. This can be achieved by using normal spatial indexing [11]. Each weighted object in Q_{wo} is input as the searching range on the data set in Q_{oi}. After these multi-level range searching, all objects of interest in both searching range and pre-defined weighted objects will be stored in the final result PQ.

When doing VRS on $WNVD$, we only compare range e with $Min(dis(q, R_n))$, $R_n \in \Re$, although which increase the size of Q_{wo} but improves the performance. Here we do not compare e with $Max(dis(q, R_n))$, because no matter whether the weighted object is fully or partially in the range, we will apply spatial index to filter the false hits in Q_{oi}. The pseudo-code of MRS is shown below:

Algorithm 2 $MRS(q, e, \Re, \wp)$

Input: query point:q, searching range:e, specified weighted object:$\Re$, objects of interest $\wp$

Output: $PQ()$

1: $Q_{wo} \longleftarrow VRS(q, e)$ on $WNVD$
2: $Q_{oi} \longleftarrow VRS(q, e)$ on NVD
3: **for** each weighted objects in Q_{wo} **do**
4: $PQ \longleftarrow$ R-tree index on Q_{oi}
5: **end for**
6: return PQ

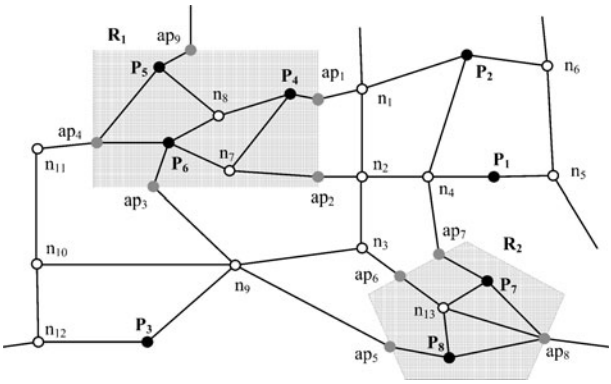


Fig. 9 An example of road networks

– Case Study

To explain the *MRS* algorithm clearer, we demonstrate the following example. The details of road networks are displayed in Fig. 9, which includes two specified weighted objects (R_1, R_2) and some objects of interest (P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8). R_1, R_2 construct a $WNVD$ (refer to Fig. 10), while P_1–P_8 construct NVD (refer to Fig. 11). Making the case simple, we jump over the detail of *VRS*, and assuming all objects shown on Fig. 9 are within the searching range. Consequently, after applying *VRS* on $WNVD$ and NVD, Q_{wo} and Q_{oi} are:

$$Q_{wo} = (R_1, R_2)$$

$$Q_{oi} = (P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8)$$

All objects of interest in Q_{oi} can be organized as a Tree structure, and then after using spatial indexing for R_1, R_2 on Q_{oi}, we can get the final result:

$$PQ() = (P_4, P_5, P_6, P_7, P_8)$$

Therein, P_4, P_5, P_6 were retrieved by R_1 and P_7, P_8 by R_2.

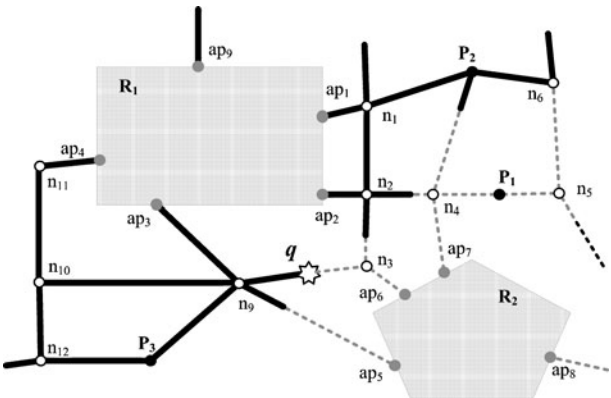


Fig. 10 Weighted network Voronoi diagram for R_1, R_2

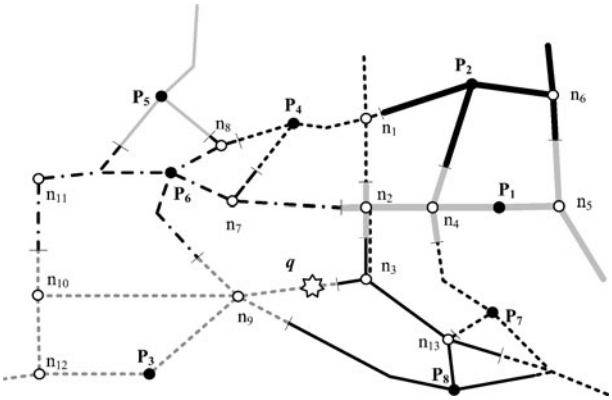


Fig. 11 Network Voronoi diagram for P_1–P_8

4 Performance evaluation

In this section, we evaluate our multi-level range search (MRS) by comparing with its main competitor, region constrained range (RCR) in low density and high density scenarios with thousands of objects in road networks.

All the experiments were programmed in Java and conducted on a IBM Thinkpad T43 laptop running MS Windows XP professional service pack 2, with an Intel Pentium Mobile CPU of 1.86 GHz, one gigabyte of RAM and 40 GB disk storage. For each algorithm, we used real-world spatial datasets with a set of objects randomly to evaluate its performance in terms of the CPU time and estimate the false hit ratio of MRS. The data of all the experiments shown below are collected by averaging the results for 1,000 random query points in each experiment to reduce the inaccuracy. The datasets of low density and high density scenarios is available in Table 2.

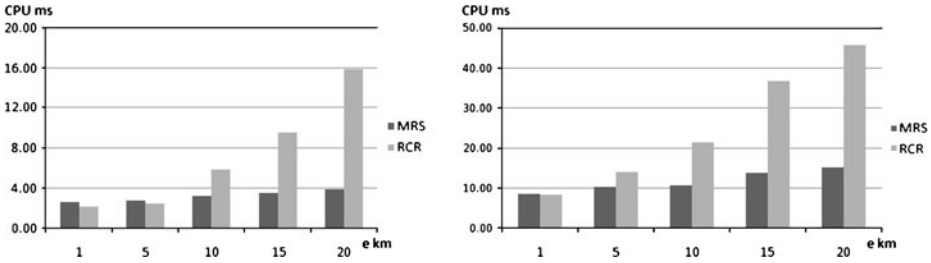
4.1 Comparison of MRS and RCR

The first set of experiments evaluate the performance of MRS and RCR in term of CPU time in low density and high density environments for both specified weighted objects (region) and objects of interest.

Figure 12 shows the processing time of our MRS and RCR in different environments relying on the density of the weighted objects (region). Since the essence of our approach is applying VRS on both WNVD and NVD, the CPU time are increasing linearly when the searching range expanding. Figure 13 illustrates the CPU time of our MRS and RCR for different density of objects of interest. The trends of MRS

Table 2 Datasets summary

	Low density scenario	High density scenario
No. of links	476	30,179
Total length (km)	20.01801	2,281.45
No. of objects	445	29,184



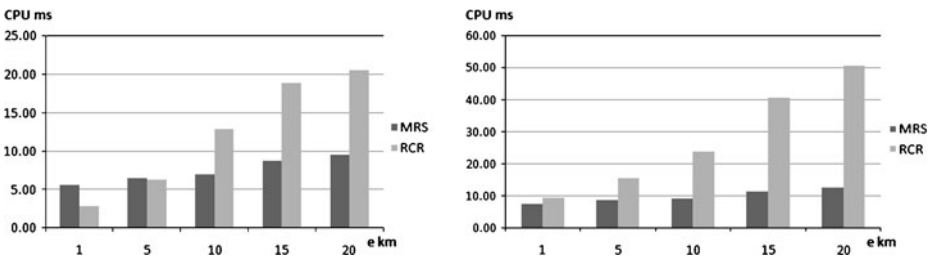
(a) CPU time in Low Density WOs (b) CPU time in High Density WOs

Fig. 12 CPU time for specified weighted objects

and RCR are similar with Fig. 12, except the CPU time increased slightly. Observing these figures, we can also see that if the searching range e is very small and the density of the corresponding objects is low, RCR has a little better performance than MRS. Because in that case, only few objects were retrieved and MRS involves some false hits. The synchrony of the MRS is not reflected clearly. But with the increasing of the searching range and the density of corresponding objects, more and more objects are involved that is main reason to decrease the performance of RCR in a high density environment, which can be observed in the last few bars in Figs. 12 and 13. Observing from the performance results, MRS outperforms RCR in most of cases.

4.2 False hit ratio of MRS

MRS retrieves some false hits, which can be filtered by finalizing $PQ()$ when doing the range search on the NVD_{oi}. So the ratio of the false hit is deserved to be tested, as shown in Fig. 14, the ratio of the false hits for MRS keeps in a relatively reasonable level.



(a) CPU time in Low Density OIs (b) CPU time in High Density OIs

Fig. 13 CPU time for objects of interest

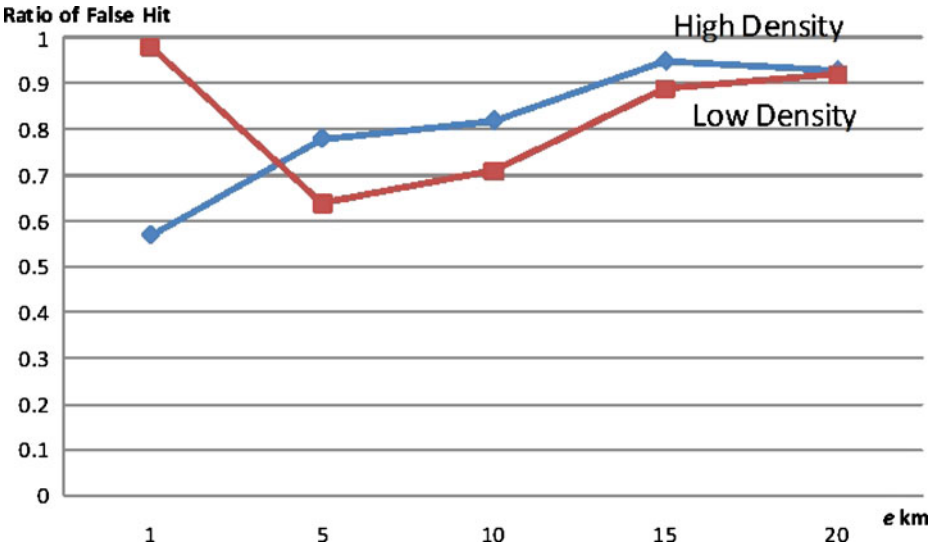


Fig. 14 Ratio of false hits for MRS

5 Related work

In this section, we briefly introduce some Voronoi-based approached on other spatial queries and see how Voronoi diagram and network Voronoi diagram can be used to improve the performance of query processing.

5.1 Voronoi-based *k* nearest neighbors

k nearest neighbor queries [25] (*k*NN) are defined as finding the nearest points of interest for the current position of user. (e.g., finding the three nearest restaurants to a office.).

– Voronoi-based *k* nearest neighbor search (VN^3)

The first approach for spatial query processing utilizing Voronoi diagram is Voronoi-based *k* nearest neighbor search (VN^3) proposed by Kolahdouzan, and Shahabi in [13]. It stores some pre-computation components of the network distances for small percentage of nodes in road networks and uses these localized data to improve the response time of the range search. In VN^3, the 1^{st} NN can be found easily by using the function *Contain*(), the generator of the polygon contains query point will be the 1^{st} NN. Then some new properties of Voronoi diagram are summarized to restrict the next nearest neighbor are within the adjacent NVPs of the previously explored nearest neighbors, which makes complexity $O(k)$ as a conservative bound of VN^3. The distance from query point to the next nearest neighbor can be calculated quickly by using the pre-computed distance. The algorithm will terminate if all *k* nearest neighbors are found. With the assistance of Voronoi diagram, VN^3 outperforms traditional range search approaches that compute the distance between every node-pair up to one order of magnitude.

– Progressive Incremental Network Expansion (PINE)

The other Voronoi-based k NN approach is progressive incremental network expansion (PINE) proposed by Safar in [18]. In summary, PINE is similar with VN^3, expect several unique attributes, which lets PINE perform better than VN^3.

After invoking the *Contain*() function, PINE performs network expansion within the polygon that contains the query points until reaches a border point. Then the next nearest neighbor is the generator of Voronoi polygon which shares the same border point with the Voronoi polygon including query point. The expansion will continue until all border points of the polygon, which contains query point, are exhausted or all k nearest neighbors are retrieved. Similar with VN^3, PINE also considers only the adjacent NVPs of the previously explored nearest neighbors as the next nearest neighbor. But differing from VN^3, PINE uses Dijkstra's algorithm to compute the distances in the polygon containing query point instead of pre-computing the distances from each border point to all the nodes inside the polygons that contain the border point off-line. Moreover PINE only accesses border-to-border (inter-cell) distances stored with the polygons rather than accessing the pre-computed border-to-border (inter-cell) and query-to-border (intra-cell) distances in VN^3. Since VN^3 requires larger amount of pre-computed values to be retrieved from the database than PINE, therefore PINE outperforms VN^3.

5.2 Voronoi-based continuous k nearest neighbors

Continuous k nearest neighbor queries (CkNN) are defined as finding the nearest points of interest along an entire path (e.g., finding the nearest gas station to a moving vehicle on any point of a pre-defined path).

– Intersection Examination (IE)

The first approach on continuous kNN is called intersection examination (IE) [14], which is based on VN^3. First of all, IE segregates the pre-defined moving path into a set of sub-segments according to the intersects of road networks. Then for each sub-segment, try to find the split point, where the result kNN need to be updated, even if there is a changes on the order of current kNN, by defining the trend (close to or far away) for each interest point in the current candidates list.

IE performs much better than other traditional kNN methods, especially when $k = 1$. For continuous 1NN queries, IE can simply find all points on a path that intersect with borders of network Voronoi diagram. In another word, the 1NN is always the generator of the current Voronoi polygon including query point.

For continuous kNN queries, IE applies segmentation on the pre-defined path, then performs VN^3 on each sub-segment to find the kNN at the two ends. The kNN result for any point on this sub-segment should be within the combination congregation of the kNN result of these two ends. In term of start point, we can get the trend of every object of interest in the kNN result, and then find the point where two adjacent objects in the kNN result get the same distance to query point. This point is defined as the split point.

– DAR/eDAR

The second continuous kNN based on Voronoi diagram is known as DAR/eDAR proposed by Safar and Ebrahimi in [19]. These algorithms are based on PINE, which

uses road networks as the underlying map. These two algorithms start with dividing the path into several sub-segments according to the network intersections and find *k*NN tables for two ends of each sub-segment, which is similar with IE. Then a pair of tables for each segments need to be compared to find out the differences and put them into two standby lists with their *k*NN. After several swaps of the objects in *k*NN result and the standby list, the *k*NNs for the two ends are same. Then every swap would be a split point. When the algorithm of DAR/eDAR is terminated, the positions of all split points will be returned to the user as well as the *k*NN result on these split points.

It is an undeniable fact that DAR and eDAR perform well for C*k*NN query than IE, since its underlying methods, PINE outperforms IE's underlying approach, VN^3.

5.3 Voronoi-based approach for other spatial queries

With the development of spatial and mobile databases system, more and more mobile queries [12, 28] are proposed. Some new methods also adopt Voronoi diagram as its underlying framework to satisfy other spatial queries, such as continuous range search [29], constrained range search, sequenced route search [20] reverse nearest neighbor and reverse farthest neighbor [24]. Most of these approaches outperform other traditional methods.

6 Conclusions

In this paper, we design a weighted Voronoi diagram as the underlying structure of our proposed *multi-level range search* (*MRS*) for region constrained range search queries in the road networks. *MRS* applies *VRS* on WNVD of weighted object and NVD of objects of interest synchronously, and then invokes a spatial indexing to filter false hits retrieved in the first step. Our experiments results show that *MRS* outperforms *RCR* in most high density and low density. The ratio of false hits are also kept in a reasonable level to ensure the efficiently of *MRS*.

In our future research, we plan to extend our algorithm to continuous range search query processing as well as detecting the moving objects [7–9, 21] of interest in the road networks. And how they can be applied in the intelligent traffic control system and context-aware system [1, 10] are also expected.

Acknowledgement This research has been partially funded by the Australian Research Council (ARC) Discovery Project (Project No: DP0987687).

References

1. Aleksy M, Butter T, Schader M (2008) Architecture for the development of context-sensitive mobile applications. Mobile Inform Syst 4(2):105–117
2. Ash PF, Bolker ED (2004) Generalized Dirichlet tessellations. Geom Dedic 20(2):209–243
3. Bayer R (1997) The universal b-tree for multidimensional indexing: general concepts. In: Proc. of worldwide computing and its applications (WWCA). Springer, New York, pp 198–209
4. Beckley DA, Evens MW, Raman VK (1985) Multikey retrieval from K-d trees and quad-trees. In: Proc. of ACM SIGMOD. ACM, New York, pp 291–301

5. Cantone D, Ferro A, Pulvirenti A, Recupero DR, Shasha D (2005) Antipole tree indexing to support range search and k-nearest neighbor search in metric spaces. IEEE Trans Knowl Data Eng 17(4):535–550
6. Dijkstra EW (1959) A note on two problems in connection with graphs. Numer Math 1(22): 269–271
7. Goh J, Taniar D (2004) Mining frequency pattern from mobile users. In: Proc. of 8th knowledge-based intelligent information and engineering systems (KES). Springer, Wellington, pp 795–801
8. Goh J, Taniar D (2005) Mining parallel patterns from mobile users. Int J Bus Data Commun Netw 1(1):50–76
9. Goh JY, Taniar D (2004) Mobile data mining by location dependencies. In: Proc. of 5th intelligent data engineering and automated learning (IDEAL). Springer, Wellington, pp 225–231
10. Gulliver SR, Ghinea G, Patel M, Serif T (2007) A context-aware tour guide: user implications. Mobile Inform Syst 3(2):71–88
11. Guttman A (1984) R-trees: a dynamic index structure for spatial searching. In: Proc. of ACM SIGMOD. ACM, New York, pp 47–57
12. Jayaputera J, Taniar D (2005) Data retrieval for location-dependent queries in a multi-cell wireless environment. Mobile Inform Syst 1(2):91–108
13. Kolahdouzan MR, Shahabi C (2004) Voronoi-based k nearest neighbor search for spatial network databases. In: Proc. of 30th VLDB. Morgan Kaufmann, Toronto, pp 840–851
14. Kolahdouzan MR, Shahabi C (2005) Alternative solutions for continuous k nearest neighbor queries in spatial network databases. GeoInformatica 9(4):321–341
15. Muhammad RB (2009) Range assignment problem on the Steiner tree based topology in ad hoc wireless networks. Mobile Inform Syst 5(1):53–64
16. Okabe A, Boots B, Sugihara K, Chiu SN (2000) Spatial tessellations: concepts and applications of Voronoi diagrams, 2nd edn. Wiley, West Sussex
17. Papadias D, Zhang J, Mamoulis N, Tao Y (2003) Query processing in spatial network databases. In: Proc. of 29th VLDB. Morgan Kaufmann, Berlin, pp 802–813
18. Safar M (2005) K nearest neighbor search in navigation systems. Mobile Inform Syst 1(3): 207–224
19. Safar M, Ebrahimi D (2006) eDAR algorithm for continuous KNN queries based on pine. Int J Inform Technol Web Eng 1(4):1–21
20. Sharifzadeh M, Shahabi C (2008) Processing optimal sequenced route queries using Voronoi diagrams. GeoInformatica 12(4):411–433
21. Taniar D, Goh J (2007) On mining movement pattern from mobile users. Int J Distrib Sensor Netw 3(1):69–86
22. Taniar D, Rahayu JW (2002) A taxonomy of indexing schemes for parallel database systems. Distrib Parallel Databases 12(1):73–106
23. Taniar D, Rahayu JW (2004) Global parallel index for multi-processors database systems. Inf Sci 165(1–2):103–127
24. Tran QT, Taniar D, Safar M (2009) Reverse k nearest neighbor and reverse farthest neighbor search on spatial networks. In: Hameurlain A (ed) Large-scale, T, data- and knowledge-centered systems, vol 1. Springer, Berlin, pp 353–372
25. Waluyo AB, Rahayu JW, Taniar D, Srinivasan B (2009) Mobile service oriented architectures for NN-queries. J Netw Comput Appl 32(2):434–447
26. Waluyo AB, Srinivasan B, Taniar D (2003) Optimal broadcast channel for data dissemination in mobile database environment. In: Proc. of 5th advanced parallel programming technologies (APPT). Springer, Xiamen, pp 655–664
27. Waluyo AB, Srinivasan B, Taniar D (2004) A taxonomy of broadcast indexing schemes for multi channel data dissemination in mobile database. In: Proc. of 18th advanced information networking and applications (AINA). IEEE Computer Society, Fukuoka, Japan, pp 213–218
28. Waluyo AB, Srinivasan B, Taniar D (2005) Research on location-dependent queries in mobile databases. Comput Syst Sci Eng 20(2):77–93
29. Xuan K, Zhao G, Taniar D, Srinivasan B, Safar M, Gavrilova M (2009) Continuous range search based on network Voronoi diagram. Int J Grid Util Comput 1(4):328–335

30. Xuan K, Zhao G, Taniar D, Srinivasan B, Safar M, Gavrilova M (2009) Network Voronoi diagram based range search. In: Proc. of 23rd advanced information networking and applications (AINA). IEEE Computer Society, Bradford, pp 741–748

Kefeng Xuan holds his Bachelor degree in Computing and Software Engineering from Monash University, Australia and Northeastern University, China in 2007, and received Master degree from Caulfield School of information technology, Monash University, Australia in 2008. He is currently a PhD candidate in Clayton School of Information technology in Monash University. His research interests include mobile computing, query processing and spatial databases.

Geng Zhao received the BS and MS degree from Caulfield School of Information Technology, Monash University, Australia in 2007 and 2008 respectively. She is currently a PhD candidate in Clayton School of Information Technology at Monash University. Her research interests include mobile computing and spatial database.

David Taniar holds Bachelor, Master, and PhD degrees—all in Computer Science, with a particular specialty in Databases. His current research interests include mobile/spatial databases, parallel/grid databases, and XML databases. He recently released a book: High Performance Parallel Database Processing and Grid Databases (John Wiley & Sons, 2008). His list of publications can be viewed at the DBLP server (http://www.informatik.uni-trier.de/~ley/db/indices/a-tree/t/Taniar:David.html). He is a founding editor-in-chief of Mobile Information Systems, IOS Press, The Netherlands. He is currently an Associate Professor at the Faculty of Information Technology, Monash University, Australia. He can be contacted at David.Taniar@infotech.monash.edu.au.

Maytham Safar is currently an Associate Professor at the Computer Engineering Department at Kuwait University. He received his Ph.D. degree in Computer Science from the University of Southern California in 2000. He has one book, three book chapters, and over sixty five conference/ journal articles. Dr. Safar!¯s current research interests include social networks, sensor networks, location based services, image retrieval, and geographic information systems. He is an IEEE and ACM Senior member. He is also a member of IEEE Standards Association, IEEE Computer Society, IEEE Geoscience & Remote Sensing Society, International Association for Development of the Information Society (IADIS), International Organization for Information Integration and Web-based Applications & Services (@WAS), and a member of the International Network for Social Network Analysis (INSNA). He is currently participating in several academic initiative programs offered by Sun Microsystems, Oracle, IBM, and Symbian. He managed to advise over twenty graduate student projects, supervised three Master Theses. He was granted over eleven research grants from research administration at Kuwait University, and Kuwait Foundation for the Advancement of Sciences (KFAS). He served on over sixty conference committees as a reviewer and/or a scientific program committee member. He also served as a member of steering committee,

organizing committee, publicity committee of over twelve conferences. He served as a reviewer for over twenty three journals. He also served as a member on the editorial board and committees on over eleven journals.

Bala Srinivasan is a professor of information technology and head of school of the Clayton School of Information Technology at the Faculty of Information Technology, Monash University, Australia. He was formerly an academic staff member of the Department of Computer Science and Information Systems at the National University of Singapore and the Indian Institute of Technology, Kanpur, India. He has authored and jointly edited 6 technical books and authored and co-authored more than 150 international refereed publications in journals and conferences in the areas of multimedia databases, data communications, data mining and distributed systems. He is a founding chairman of the Australasian database conference. He was awarded the Monash Vice-Chancellor medal for post-graduate supervision. He holds a Bachelor of Engineering Honours degree in electronics and communication engineering, a master and a PhD degree, both in computer science.