Distance-Based Location Update and Routing in Irregular Cellular Networks

Victor Chepoi[1], Feodor F. Dragan[2], and Yann Vaxès[1]

[1] Laboratoire d'Informatique Fondamentale de Marseille, Université de la Méditerranée, Faculté des Sciences de Luminy, F-13288 Marseille Cedex 9, France

{chepoi,vaxes}@lif.univ-mrs.fr

[2] Department of Computer Science, Kent State University, Kent, Ohio 44242, USA

dragan@cs.kent.edu

Abstract. In this paper, we consider a class of cellular networks, called irregular cellular networks, which may have a non-uniform distribution of base stations and a non-uniform cell size. The communication (between base stations) graph of such a cellular network forms a so-called trigraph, i.e., a plane triangulation with inner vertices of degree at least six. We show that each trigraph with n vertices admits a labeling that assigns $O(\log^2 n)$ bit labels to vertices of the graph such that the distance between any two vertices u and v can be determined in constant time by merely inspecting the labels of u and v, without using any other information about the graph. Furthermore, we show that there is a labeling, assigning labels of size $O(\log^2 n)$ bits to vertices, which allows, given the label of a source vertex and the label of a destination, to compute in constant time the port number of the edge from the source that heads in the direction of the destination. These two results for trigraphs provide elegant solutions to a few problems in irregular cellular networks. The distance labeling scheme allows efficient implementation of the distance-based tracking protocol, by providing information, generally not available to the user, and means for accurate cell distance determination. Our routing and distance labeling schemes provide compact and efficient routing and connection re-routing protocols. Although these results are primarily developed for cellular networks, they may find applications also in other types of wireless networks that have a fixed backbone infrastructure.

Keywords: irregular cellular networks, plane triangulations, cell distance, location management, routing, efficient algorithms.

1 Introduction

Cellular communications have experienced an explosive growth recently. In a *cellular network*, a service area is divided into smaller areas called *cells*. Each cell is served by a *base station* (BS) to which mobile users must connect to make or receive phone calls. Mobile users are normally connected to the nearest BSs and, thus, the BSs divide the service area such that each BS serves all users that are located inside a cell centered at BS. Cellular designs often use the hexagonal configuration because it is able to cover the maximum area with the minimum number of BSs [5, 17]; hexagons fit together without any overlap or gap in between them and their shape approaches a circular shape which is the ideal power coverage area. In a such designed cellular network, two BSs can directly communicate with each other if the corresponding hexagons are neighbor cells. Let formally define a *hexagonal system* as a subgraph of the regular hexagonal grid which is formed by a simple circuit of the grid and the region bounded by this circuit. Then communications between the BSs go via the dual graph of a hexagonal system, a so called *triangular system*. The vertices of this triangular system are the BSs and two BSs are connected by an edge if and only if the corresponding cells (hexagons) share a common boundary (see Fig. 1 for an illustration). All triangular systems are connected subgraphs of the triangular grid.

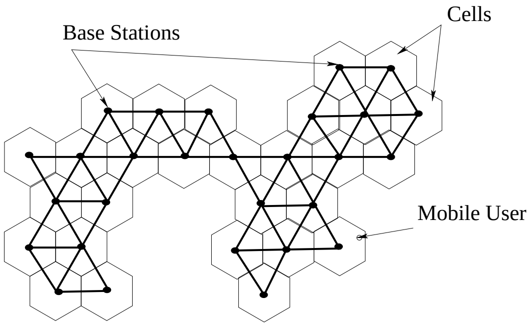

Fig. 1. A cellular network modeled by a triangular system.

Mobile users with cellular phones have to register frequently to facilitate their location when phoning them. They move from cell to cell, but do not always contact their new BS to update their position since too many messages may be required and the system may be blocked for regular calls. In [3], Bar-Noy et al. proposed three dynamic location update (or registration) schemes: *time-based, movement based, and distance-based*. It has been shown that the distance-based scheme is the most efficient among the three [3]. In the distance-based location update scheme, a mobile user keeps track of the cell distance between the current cell and the last reported cell. Here, the *cell distance* is the number of cells on a shortest route between the two cells (i.e., it is the graph distance between the corresponding BSs computed in the triangular system). When the cell distance reaches a predefined threshold, say D, the mobile user updates its location. The cell distance based strategy guarantees that the mobile user is located in the area that is within a cell distance D from the last reported cell. When an incoming call arrives for a mobile user, the cellular system will page all the cells within a distance of D from the last reported cell. The cell distance based strategy is dynamic, and the distance threshold D can be determined on a per-user basis depending of his/her mobility patterns.

Thus, additional to efficient routing protocols for cellular networks, there is a need in efficient computation of the cell distance between any two cells. However, it has been claimed in [3] that it is hard to compute the distance between two cells, or it requires a lot of storage to maintain the distance information among all cells [1, 13]. Current cellular networks do not provide information that can be used to derive cell distances. In [14], Nocetti et al. proposed a very simple method to compute the cell distance between any two cells in the case when a cellular network is modeled by an *isometric subgraph* of the triangular grid (that is, the distance between any two vertices in the subgraph coincides with the distance in the triangular grid; see Fig. 2). The distance computation of [14] is based on a new cell addressing scheme which provides also a short and elegant routing protocol.

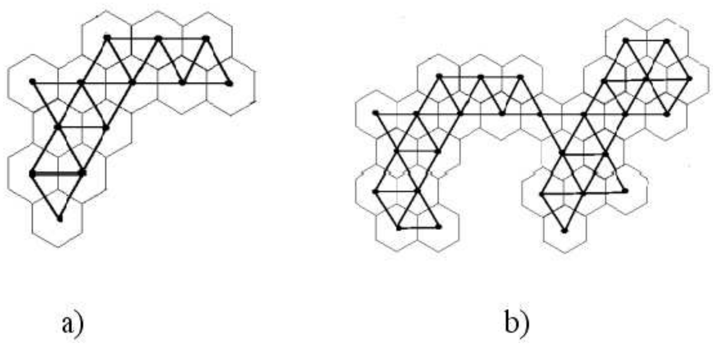

Fig. 2. a) A cellular network modeled by an isometric (i.e., a distance preserving) subgraph of the triangular grid. b) A cellular network modeled by a general triangular system. The distance between two lowest non-adjacent vertices in the system is 8 while the distance between them in the full triangular grid would be 3 (or 4).

Unfortunately, the analytic method from [14], works only for those triangular systems which are the isometric subgraphs of the triangular grid. Recently in [7], we proposed an addressing scheme for arbitrary triangular systems by employing their isometric embeddings into the Cartesian product of three trees. This embedding provides a simple representation of any triangular system with only three small integers per vertex, and allows to employ the compact labeling schemes for trees for distance queries and routing. We have shown that each such system with n vertices admits a labeling that assigns $O(\log^2 n)$ bit labels to vertices of the system such that the distance between any two vertices u and v can be determined in constant time by merely inspecting the labels of u and v, without using any other information about the system. This provides, for example, the following compact and efficient dynamic location update scheme. Each mobile user stores locally, using only $O(\log^2 n)$ bits, the label of the last base station it was registered with. Upon receiving the $O(\log^2 n)$-bit label of a base station in cell of which it is currently located, it computes in constant time the cell distance between the two base stations and updates location if the cell distance exceeds a predefined threshold. Note that for a cellular network with at most 10000 base stations, our label sizes will not exceed 353 bits. Additionally, we have shown in [7] that there is also a labeling, assigning labels of size $O(\log n)$ bits to vertices of a triangular system, which allows, given the label of a source vertex and the label of a destination, to compute in constant time the port number of the edge from the source that heads in the direction of the destination. This provides an elegant and efficient routing protocol for a cellular network modeled by an arbitrary triangular system.

Irregular cellular networks and the results of this paper. Although a uniform configuration of BSs (which results in uniformly-sized, hexagonal shaped cells and in a triangular system type connection between BSs) is always used in the planning, the final base station placement may not be uniformly distributed because in reality an obstacle such as a river (a lake, a historic site, etc.) prevents a base station from being placed at the planned location. Moreover, in order to accommodate more subscribers, cells of a previously deployed cellular network need to be split or rearranged into smaller ones to make more efficient use of the limited frequency spectrum allocated [5, 17]. As a result, the cell size in one area may be different from the cell size in another area. For instance, in a dense-populated area of a city the cell size may be smaller than the cell size in a rural area.

Very little is known about the cellular networks with non-uniform cell sizes or a non-uniform distribution of base stations (we call them *non-uniform cellular networks*). Only recently, some of these issues have been partially addressed in papers [10, 11]. The authors of [11] consider the problem of placing a newly planned cellular network over a set of existing BSs in such a way that the maximum number of old BSs can be reused (an existing BS should be put as close to a cell center as possible in the planned cellular network). They assume that the planned network is a uniform one, while the existing BSs are arbitrarily distributed. To compute the cell distance between two cells in a cellular network with non-uniformly distributed BSs, paper [10] proposes to design an optimal virtual hexagonal system, with a uniform virtual cell size, such that each virtual cell will contain at most one base station. Then, an address (a label) can be assigned to a BS based on the position of the BS in the virtual hexagonal system, and the distance between two BSs can be computed using methods from [14, 7]. Unfortunately, [10] does not elaborate on how close the cell distance in the virtual hexagonal system constructed will approximate the cell distance in the original cellular network.

In this paper, we make one step further in analyzing non-uniform cellular networks. We do not require from BSs to be set in a very regular pattern. Our only requirement is that cells resulted from this BS deployment should obey the following property: *each inner cell (cell which is not incident to the service area boundary) has at least six neighbor cells* (a property naturally fulfilled by the hexagonal systems, where each inner cell has exactly six neighbor cells). These non-uniform cellular networks will be called *irregular cellular networks* (see Fig. 3 for an illustration). Irregular cellular networks give a more flexible way of designing a cellular system.

In a cellular network, the area covered by a BS can be viewed as the Voronoi polygon [16] of the BS. Let assume that a set $S = \{P_1, \ldots, P_n\}$ of n BSs is arbitrarily positioned in the Euclidean plane and a service area they cover is given by a polygonal region R. Then the Voronoi polygon of a BS P_i will consist of all points of R which are closer to P_i than to any other points of S, i.e., it is a polygon $VP(P_i) := \{p \in R : \rho(p, P_i) \leq \rho(p, S)\}$ [16]. These polygons divide the service area R into n cells, each of

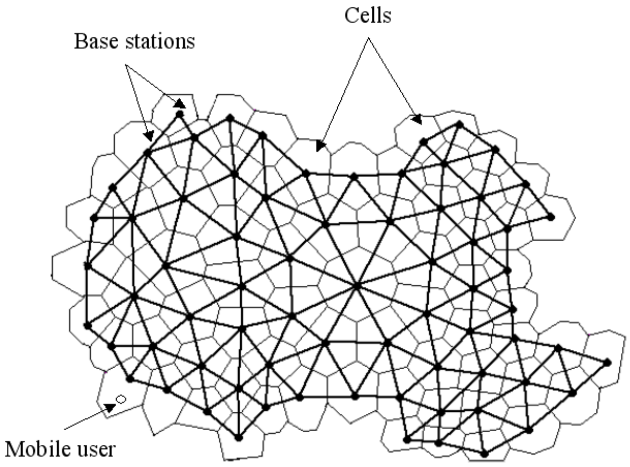

Fig. 3. An irregular cellular network modeled by a trigraph.

which is associated with a BS from S. We say that a cell is an *inner cell* if it is not incident to the boundary of the service area R, and two cells are *neighbor cells* if the corresponding polygons share a common side. Define the communication (dual) graph of this cellular system as follows: the vertices of this graph are the BSs and two BSs are connected by an edge if and only if the corresponding cells are neighbor cells (see Fig. 3). Thus, the communication graph is simply the geometric dual of the Voronoi tesselation, whence it is nothing else but the Delaunay graph of the set S [16]. In particular, the communication graph of an irregular cellular network is a plane triangulation with all inner vertices of degree at least 6. These kind of plane triangulations were called *trigraphs* in [6].

In this paper we show that each trigraph with n vertices admits a labeling scheme that assigns $O(\log^2 n)$ bit labels to vertices of the graph such that the distance between any two vertices u and v can be determined in constant time by merely inspecting the labels of u and v, without using any other information about the graph. This distance labeling scheme allows efficient implementation of distance-based tracking protocol in any irregular cellular network, by providing information, generally not available to the user, and means for accurate cell distance determination. Additionally, we show that there is also a labeling scheme, assigning labels of size $O(\log^2 n)$ bits to vertices of a trigraph, which allows, given the label of a source vertex and the label of a destination, to compute in constant time the port number of the edge from the source that heads in the direction of the destination. This provides an elegant and efficient routing and connection re-routing protocols for all irregular cellular networks.

Note that these results can find applications also in other types of *wireless networks having a fixed backbone infrastructure*. In a related paper [8], using a more sophisticated analysis, among other results we present compact and efficient labeling schemes for more general class of graphs, namely, for all plane graphs with all inner faces of length at least 3 and with all inner vertices of degree at least 6 (so called *(3,6)-graphs*). Based on geometric properties of those graphs, we design distance and routing labeling schemes with labels of size $O(\log^2 n)$ bits per vertex and constant time distance decoder and routing decision.

2 Geometric properties of trigraphs

In this section, we provide all necessary definitions and the geometric properties (named (P1)-(P4)) of trigraphs on which our distance and routing labeling schemes are based. Most of these properties have been established in [8] in a more general context; see also [2] for other related properties of trigraphs. All graphs $G = (V, E)$ occurring in this paper are trigraphs, i.e., undirected, unweighted, connected, n-vertex *plane graphs* [16, 18] such that all inner faces are triangles and all inner vertices have degree at least 6. The *distance* $d(u, v) := d_G(u, v)$ between two vertices u and v is the length of a shortest (u, v)-path. For a vertex u and a subset of vertices S, the *distance* from u to S is $d(u, S) = \min\{d(u, s) : s \in S\}$. An induced subgraph of G (or the corresponding vertex set) is called *convex* if it includes all shortest paths between any of its vertices. For a set $S \subseteq V$ and a vertex x of G, the *projection* $Pr(x, S)$ of x on S consists of all vertices $v \in S$ at minimum distance from x. Notice that, for a convex set $S \subseteq V$ and any pair of vertices $s \in S$ and $x \in V - S$, there is a shortest path from x to s passing through $Pr(x, S)$.

Two neighbors x, y of a vertex v of G are called *consecutive* if v, x, y belong to a common inner face of G. For an edge uv of a graph G, define the following partition of the vertex set V:

$$W(u, v) := \{x \in V : d(x, u) < d(x, v)\},$$

$$W(v, u) := \{x \in V : d(x, v) < d(x, u)\},$$

$$W_=(uv) := \{x \in V : d(x, v) = d(x, u)\}.$$

For a graph $G = (V, E)$ and a vertex x, let $F(x) := \sum_{v \in V} d(x, v)$. Any vertex minimizing the function F is called a *median vertex* of the graph G. Notice the following simple but important property of the function F : *if uv is an edge of G, then* $F(v) - F(u) = |W(u, v)| - |W(v, u)|$. From this we immediately conclude that if v is a median vertex of G, then $|W(u, v)| \le |V|/2$ for any neighbor u of v. A linear-time algorithm for computing medians of trigraphs is proposed in [9].

A *cut* $\{A, B\}$ of G is a partition of the vertex-set V into two parts, and a *convex cut* is a cut in which the halves A and B are convex. Denote by $E(A, B)$ the set of all edges of G having one end in A and another one in B, and say that those edges are *crossed* (or *cut*) by $\{A, B\}$. The *zone* $Z(A, B)$ of the cut $\{A, B\}$ is the family of inner faces (triangles) of G sharing edges with $E(A, B)$ (see Fig. 4 for an illustration). A zone $Z(A, B)$ is called a *strip* if the faces of $Z(A, B)$ induce a simple path in the dual graph of G and two faces F', F'' of $Z(A, B)$ intersect if and only if they share an edge of $E(A, B)$. We will use the same notation $Z(A, B)$ for the (plane) subgraph of G induced by the vertices and the edges occurring in the faces of this zone. The inner faces of this graph are exactly the inner faces of G from $Z(A, B)$. For a strip $Z(A, B)$, call the subgraphs induced by $\partial A := Z(A, B) \cap A$ and $\partial B := Z(A, B) \cap B$ the *border lines* of the cut $\{A, B\}$.

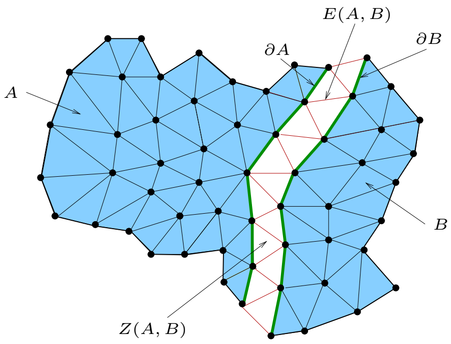

Fig. 4. A *convex* cut $\{A, B\}$, its zone $Z(A, B)$, its border lines ∂A and ∂B and the set $E(A, B)$ of edges crossed by this cut.

We continue with the definition of an *alternating cut*. Suppose that all inner faces (triangles) of G are oriented clockwise. An edge e of a triangular face F has two opposite edges, a left opposite edge e^+ and a right opposite edge e^-. We say that the cut $\{A, B\}$ makes *a right or a left turn* on a face $F \in Z(A, B)$ depending on which of the pairs $\{e, e^+\}$ or $\{e, e^-\}$ it crosses. A cut $\{A, B\}$ of a plane triangulation G is *alternating* if the turns on it alternate. As was shown in [2, 8], the following property holds for all trigraphs.

(P1) *The alternating cuts of trigraphs are all convex, their border lines are convex paths and their zones are strips sharing two edges with ∂G, the boundary of the plane graph G.*

As a consequence, every edge of G is crossed by exactly two alternating cuts. Notice also, that

(P2) *The projection of a vertex x on the zone of an alternating cut is a convex path whose vertices have the same distance to x.*

Thus, information about this projection of vertex x can be compactly represented by the end vertices of this path and the distance from x to the projection. Also, for routing messages from x via the zone $Z(A, B)$ of an alternating cut, the following property is crucial.

(P3) *Either x has a neighbor which is one step closer to $Z(A, B)$ and whose projection on this zone coincides with that of x, or there exist two neighbors u_x and v_x of x one step closer to the zone and whose projections on $Z(A, B)$ cover the projection of x.*

We will keep at x the information about such neighbors and use it in the routing decision.

Let $e = xy$ be an edge of ∂G and let $\{A', B'\}$ and $\{A'', B''\}$ be the (not necessarily distinct) alternating cuts crossing e, where $x \in A' \cap A''$ and $y \in B' \cap B''$. We will specify now a relation between A', B', A'', B'' and the sets $W(x, y), W(y, x), W_=(xy)$. By removing the edges of $E(A', B') \cup E(A'', B'')$ from G but leaving their end vertices, we get a graph whose connected components are induced by the pairwise intersections $A' \cap A'', B' \cap B'', A' \cap B''$, and $A'' \cap B'$. It is established in [8] that these convex sets coincide with $W(x,y), W(y,x)$ and the connected components of $W_=(xy)$. More precisely, the following holds: $W(x,y) = A' \cap A'', W(y,x) = B' \cap B''$, while $W_=(x,y) := B' \cap A''$ and $W_=(y,x) := A' \cap B''$ constitute a partition of $W_=(xy)$ into two (maybe empty) convex subsets (see Fig. 5a).

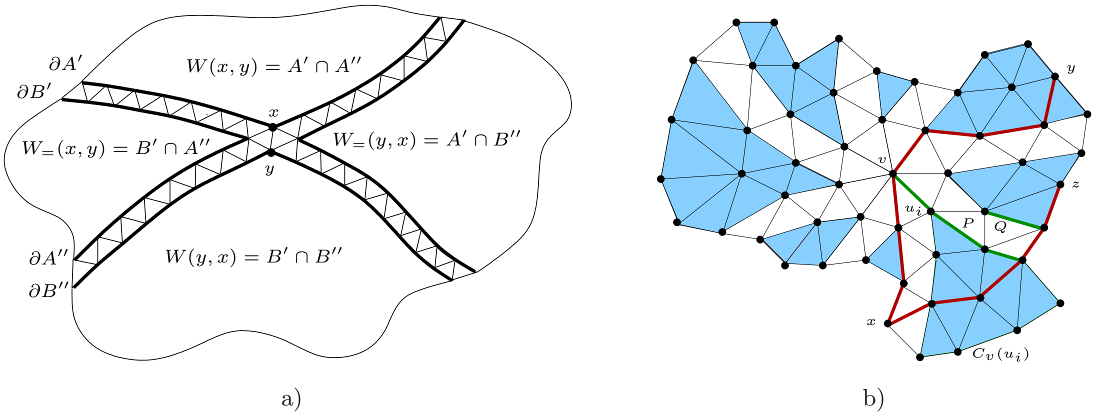

Fig. 5. a) The pair of alternating cuts crossing the edge $e = xy$; b) The partition into cones around the vertex v. Since x and z lie in 2-neighboring cones, we have $d(x,z) = d(x,P) + d(P,Q) + d(Q,z)$. On the other hand, x and y lie in 3-neighboring cones implying $d(x,y) = d(x,v) + d(v,y)$.

Finally, the following *partition into cones* of a trigraph G will be very useful for our labeling schemes. Let v be a median vertex of G, called the *pivot of the partition*, and let $u_0, u_1, \ldots, u_{k-1}$ be its neighbors in counterclockwise order around v, according to the embedding of G in the plane. Every edge vu_i is crossed by two alternating cuts $\{A'_i, B'_i\}$ and $\{A''_i, B''_i\}$ such that $v \in A'_i \cap A''_i$ and $u_i \in B'_i \cap B''_i$. Let us orient the cuts $\{A'_i, B'_i\}$ and $\{A''_i, B''_i\}$ such that v is on the left border line of each of them. In this case, we will denote by $\{A'_i, B'_i\}$ that cut from the two alternating cuts separating u_i and v, such that the last turn before u_iv is on the right and the next turn after u_iv is on the left. For each vertex u_i, set $C_v(u_i) = B'_i \cap A'_{i+1(mod\ k)}$ and call $C_v(u_i)$ a *cone with apex* u_i. Every cone is convex as the intersection of two convex sets. Since $C_v(u_i) \subseteq W(u_i, v)$ and v is a median vertex, we obtain $|C_v(u_i)| \leq n/2$. Furthermore, the cones $C_v(u_0), \ldots, C_v(u_{k-1})$ together with the vertex v form a partition of the vertex-set of G (see Fig. 5b). To report the distance or a routing path between two query vertices x and y efficiently, yet another property of the partition $C_v(u_0) \cup \ldots \cup C_v(u_{k-1}) \cup \{v\}$ is significant. We call two neighbors u_i, u_j of v *p-consecutive* and their cones $C_v(u_i), C_v(u_j)$ *p-neighboring* if $\min\{|i-j|, k-|i-j|\} = p$.

(P4) *Let $x \in C_v(u_i)$ and $y \in C_v(u_j)$. If $C_v(u_i)$ and $C_v(u_j)$ are not p-neighboring for $p \leq 2$, then $d(x,y) = d(x,v) + d(v,y)$.*

3 An overview of the method

Our distance and routing labeling schemes are based on the geometric properties (P1)-(P4) of trigraphs presented in the previous section and use a hierarchical subdivision of the input trigraph G into cones. Namely, let v be a median vertex of G and let $C_v(u_0) \cup \ldots \cup C_v(u_{k-1}) \cup \{v\}$ be the partition of G into cones (as defined in previous section). We recursively subdivide into cones the subgraph G_i induced by the cone $C_v(u_i)$ for $i = 0, 1, 2, \ldots, k-1$. This suggests the necessity of building a decomposition tree $T(G)$ of G, by taking pair (G, v) to be the root and connecting the root of each tree $T(G_i)$ as a child of (G, v). It is easy to see that a decomposition tree $T(G)$ of a graph G with n vertices has depth at most $\log_2 n$ and can be constructed in $O(n \log n)$ time. Indeed, in each level of recursion we need to find median vertices of current subgraphs (which can be done in linear time [9]) and to construct the corresponding cones. Also, since the graph sizes are reduced by a factor 1/2, the recursion depth is $O(\log n)$.

Every vertex $x \in V$ belongs to at most $O(\log_2 n)$ cones in this subdivision. In order to compute the distances from x to any other query vertex y or to route a message arriving in x, we keep in x the distances from x to the pivots and to the zones of alternating cuts defining these cones, as well as a compact (thanks to property (P2)) representation of the metric projections of x on these zones. Now, given x and y, in order to report the distance $d(x, y)$, we compute the highest level of the subdivision such that x and y lie in different cones. This can be done in constant time using the labeling scheme for depths of nearest common ancestors (*NCA-depth labeling scheme*) described below. If the respective cones are neither 1- nor 2-neighboring then, according to (P4), the distance $d(x, y)$ is simply the distance from x to the corresponding pivot v plus the distance from the pivot to y, i.e., $d(x, y) = d(x, v) + d(v, y)$. Otherwise, if the cones containing x and y are 1 or 2-neighboring, then, according to (P1), $d(x, y)$ is obtained as the sum of the distances from x and from y to their respective projections P and Q on the zone of an alternating cut separating them plus the distance between these two projections, i.e., $d(x, y) = d(x, P) + d(P, Q) + d(Q, y)$ (see Fig. 5b). As it will be shown in Section 3, the simple structure of an alternating cut allows to compute the distance between two projections on its zone in $O(1)$ time.

Routing between x and y can be performed by converting the distance labeling scheme in the following way. To route a message from x to y lying in different cones, additional to distances, we have to store in the label of x the output port number of the first edge on a shortest path from x to the pivot and, according to (P3), the output port number of the first edge on a shortest path from x to each of the two end vertices of the projection of x on the 1-neighboring as well as 2-neighboring cones, or more precisely on the zones separating the respective cones. If x is its own projection on the zone between x and y, we consider the relative position of x and the projection of y on the zone to decide in constant time via which edge the message should be sent.

As it was mentioned above, for the tree $T(G)$ we need a labeling scheme for depths of nearest common ancestors (*NCA-depth labeling scheme*). In [15] such a scheme with $O(\log^2 n)$ bit labels and with $O(\log n)$ query time was presented for any tree with n nodes. The scheme assigns $O(\log^2 n)$ bit labels to nodes of the tree such that the depth of the nearest common ancestor of any two query nodes can be determined in $O(\log n)$ time by merely inspecting their labels, without using any other information about the tree. Here, we can use the fact that $T(G)$ has the $O(\log n)$ depth and get constant query time in this case. To do this one can simply translate the technique of Harel and Tarjan [12] to a labeling scheme. Note that whenever they access global information, it is associated with an ancestor in a tree. Since the depth of our tree is $O(\log n)$, one can copy this ancestor information down to each descendant and get the desired label of $O(\log^2 n)$ bits. Thus, tree $T(G)$ can be preprocessed in $O(n \log n)$ time for depths of nearest common ancestors. This preprocessing step creates for $T(G)$ an NCA-depth labeling scheme with $O(\log^2 n)$ bit labels and constant query time.

4 Distance and routing labeling schemes

4.1 Computing the distance between $x \in A$ and $y \in B$

In this section, we establish how to compute the distance $d(x, y)$ between two vertices $x \in A$ and $y \in B$, where $\{A, B\}$ is an alternating cut with the zone $Z(A, B)$. We will use the short-hands $P := Pr(x, Z(A, B))$ and $Q := Pr(y, Z(A, B))$. The distance between x and y can be computed using the formula $d(x, y) = d(x, P) + d(P, Q) + d(y, Q)$ whose correctness is provided by (P1).

Pick an edge $ab \in E(A, B) \cap \partial G$, where $a \in \partial A$, $b \in \partial B$, and ∂A is the left border line of $Z(A, B)$. Suppose, without loss of generality, that the last turn of $E(A, B)$ before the edge ab is to the right (the other case being analogous). Due to the simple structure of the zone of an alternating cut presented in Section 2, the distance between two vertices $p \in \partial A$ and $q \in \partial B$ is

$$d(p, q) = \begin{cases} d(q, b) - d(p, a) + 1 & \text{if } d(p, a) \leq d(q, b) \\ d(p, a) - d(q, b) & \text{if } d(p, a) > d(q, b). \end{cases} \tag{1}$$

Also, the distance between a subpath P of ∂A having p', p'' as end-vertices (with $d(p', a) \leq d(p'', a)$) and a subpath Q of ∂B having q', q'' as end-vertices (with $d(q', b) \leq d(q'', b)$) can be computed in constant time from the distances $d(p', a), d(p'', a), d(q', b), d(q'', b)$, by using the following formula (see Fig. 6):

$$d(P, Q) := \begin{cases} d(p'', q') & \text{if } d(p'', a) \leq d(q', b) \\ d(p', q'') & \text{if } d(p', a) > d(q'', b) \\ 1 & \text{otherwise.} \end{cases} \tag{2}$$

In view of these formulas, the function $\mathsf{distance\_graphs}$ defined below can retrieve $d(x, y)$ in $O(1)$ time from the label

$$D_x := (\overset{1}{1},\ \overset{2}{d(x, P)},\ \overset{3}{d(p', a)},\ \overset{4}{d(p'', a)})$$

of $x \in A$ and the label

$$D_y := (\overset{1}{0},\ \overset{2}{d(y, Q)},\ \overset{3}{d(q', b)},\ \overset{4}{d(q'', b)})$$

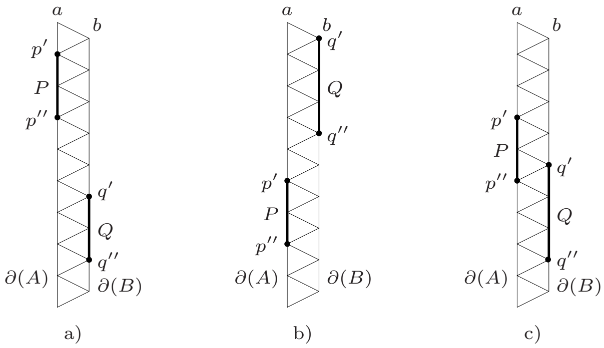

Fig. 6. Illustration to formulas (1) and (2): a) $d(P,Q) = d(p'',q') = d(b,q') - d(a,p'') + 1$; b) $d(P,Q) = d(p',q'') = d(a,p') - d(b,q'')$; c) $d(P,Q) = 1$.

of $y \in B$ where p', p'' are the end vertices of P and q', q'' the end-vertices of Q. The first entry in D_x (and in D_y) is a bit that indicates if the last turn of $E(A,B)$, before the edge ab, is to the right. If the last turn is to the left, we need to interchange the roles of x and y in the consideration.

function distance_graphs(D_x, D_y)

if $D_x(1) = 0$ then /* rename inputs */ set $C := D_x, D_x := D_y, D_y := C$
if $D_x(4) \leq D_y(3)$ then
  return $D_x(2) + (D_y(3) - D_x(4) + 1) + D_y(2)$
else
  if $D_x(3) > D_y(4)$ then
    return $D_x(2) + (D_x(3) - D_y(4)) + D_y(2)$
  else
    return $D_x(2) + 1 + D_y(2)$

4.2 Distance decoder

Here we explain how, using the decomposition tree $T(G)$, one can find the distance between any two vertices of G. First, we will describe the labels of vertices of G.

Let v be a median vertex of G (which we assume to be an inner vertex), and let $u_0, \ldots, u_{k-1}$ be its neighbors in counterclockwise order around v. Recall that the cones $C_v(u_i)$, $i \in \{0, \ldots, k-1\}$ of G were defined as follows: $C_v(u_i) = B'_i \cap A'_{i+1(mod\ k)}$. Each vertex $y \in V \setminus C_v(u_i)$ is separated from a vertex $x \in C_v(u_i)$ by zone $Z(A'_i, B'_i)$ or by zone $Z(A'_{i+1}, B'_{i+1})$. From (P1) we know that, if two vertices x and y lie in two 1-neighboring or 2-neighboring cones, then $d(x,y)$ is realized via their projections on the zone separating these cones, and, from (P4), if x and y belong to p-neighboring cones with $p > 2$, then $d(x,y)$ is realized via v. For any vertex $x \in C_v(u_i)$ and index $j = i, i+1, i+2\ (mod\ k)$, let D^j_x be the distance label of x with respect to the cut $\{A'_j, B'_j\}$ (D_x was defined in the previous subsection with respect to an arbitrary cut $\{A, B\}$). Let also G_i be a subgraph of G induced by $C_v(u_i)$ ($i \in \{0, \ldots, k-1\}$).

Assume that a decomposition tree $T(G)$ of G and its NCA-depth labeling scheme are given. For a vertex x of G, let $S(x)$ be the deepest node of $T(G)$ containing x and A_x be the label of $S(x)$ in the NCA-depth labeling scheme. Let also $S_0, S_1, \ldots, S_h$ be the nodes of the path of $T(G)$ from the root (G, v) (which is S_0) to the node $S(x) = S_h$.

In the distance labeling, the label $L(x)$ will be the concatenation of A_x, and $h+1$ tuples $\tau_0^x, \tau_1^x, \ldots, \tau_h^x$ where τ_q^x ($q \in \{0, \ldots, h\}$) is defined as follows. Let S_q be a node (G_q, v_q) of $T(G)$. Assume that x belongs to a cone $C_{v_q}(u_i)$ of G_q for some $i \in \{0, ..., \delta_{G_q}(v_q) - 1\}$ (here $\delta_{G_q}(v_q)$ is the degree of vertex v_q in the graph G_q). Then,

$$\tau_q^x := (\overset{1}{i},\ \overset{2}{\delta_{G_q}(v_q)},\ \overset{3}{d_{G_q}(x, v_q)},\ \overset{4}{D_x^i},\ \overset{5}{D_x^{i+1}},\ \overset{6}{D_x^{i+2}}),$$

where zones and projections are considered in graph G_q. If $x = v_q$, we set $\tau_q^{v_q} := (\delta_{G_q}(v_q), \delta_{G_q}(v_q), 0, 0, 0, 0)$.

Since the depth of $T(G)$ is $O(\log n)$, $L(x)$ is of length $O(\log^2 n)$ bits for any $x \in V$. Note that computation of those tuples can be incorporated into the algorithm of building $T(G)$, leading to an $O(n \log n)$ time computation of all labels $L(x), x \in V$ (for a graph G_q, the paths $Pr(x, Z(A'_j, B'_j))$ ($j = i, i+1, i+2$) and corresponding distances can be computed by running Bread-First-Searches from v_q and $Z(A'_j, B'_j)$ ($j = i, i+1, i+2$)).

Algorithm DISTANCE_DECODER: Distance decoder for trigraphs.

Input: two labels $L(x) = A_x \circ \tau_0^x \circ \tau_1^x \circ \cdots \circ \tau_h^x$ and $L(y) = A_y \circ \tau_0^y \circ \tau_1^y \circ \cdots \circ \tau_q^y$.

Output: $d(x, y)$, the distance between x and y in G.

Method:

use A_x and A_y to find the depth l in $T(G)$ of
  the nearest common ancestor of $S(x)$ and $S(y)$;
extract from $L(x)$ and $L(y)$ the tuples τ_l^x and τ_l^y;
if $\tau_l^x(1) = \tau_l^x(2)$ then output $\tau_l^y(3)$ and stop; /* $x = v_q$ */
if $\tau_l^y(1) = \tau_l^y(2)$ then output $\tau_l^x(3)$ and stop; /* $y = v_q$ */
/* if the cones are 1-neighboring */
if ($\tau_l^x(1) = \tau_l^y(1) - 1$ or $\tau_l^y(1) = 0$ and $\tau_l^x(1) = \tau_l^x(2) - 1$) then output
  distance_graphs($\tau_l^x(5), \tau_l^y(4)$) and stop;
if ($\tau_l^y(1) = \tau_l^x(1) - 1$ or $\tau_l^x(1) = 0$ and $\tau_l^y(1) = \tau_l^x(2) - 1$) then output
  distance_graphs($\tau_l^y(5), \tau_l^x(4)$) and stop;
/* if the cones are 2-neighboring */
if ($\tau_l^x(1) = \tau_l^y(1) - 2$ or $\tau_l^y(1) = 0$ and $\tau_l^x(1) = \tau_l^x(2) - 2$ or
  $\tau_l^y(1) = 1$ and $\tau_l^x(1) = \tau_l^x(2) - 1$) then output

distance_graphs$(\tau_l^x(6), \tau_l^y(4))$ and stop;

if $(\tau_l^y(1) = \tau_l^x(1) - 2$ or $\tau_l^x(1) = 0$ and $\tau_l^y(1) = \tau_l^x(2) - 2$ or $\tau_l^x(1) = 1$ and $\tau_l^y(1) = \tau_l^x(2) - 1)$ then output distance_graphs$(\tau_l^y(6), \tau_l^x(4))$ and stop;

else output $\tau_l^x(3) + \tau_l^y(3)$.

4.3 Routing from $x \in A$ to $y \in B$

From (P3), we know that any vertex $x \in A$ has one or two neighbors v_x and u_x such that $I(x,y) \cap \{v_x, u_x\} \neq \emptyset$ for any vertex $y \in B$ (here, $I(x,y) := \{v \in V : d(x,v) + d(v,y) = d(x,y)\}$). Thus, the message from x should be forwarded to that of these neighbors which is closer to y. If $x \in A \setminus \partial A$, then $u_x, v_x \in A$ and this decision can be made in $O(1)$ time by decoding the distances $d(y, v_x)$ and $d(y, u_x)$. Define $help(v_x)$ to be equal to 1 if x and v_x are separated by the cut $\{A, B\}$ and 0 otherwise ($help(u_x), help(v_y), help(u_y)$ are defined analogously). Then, in this case we can make a routing decision in $O(1)$ time from the label

$$R_x := (\overset{1}{D_x}, \overset{2}{D_{v_x}}, \overset{3}{D_{u_x}}, \overset{4}{port(x, v_x)}, \overset{5}{port(x, u_x)}, \overset{6}{help(v_x)}, \overset{7}{help(u_x)})$$

of x and the label D_y of $y \in B$ (vice versa, to route from $y \in B \setminus \partial B$ to $x \in A$ we need the label

$$R_y := (\overset{1}{D_y}, \overset{2}{D_{v_y}}, \overset{3}{D_{u_y}}, \overset{4}{port(y, v_y)}, \overset{5}{port(y, u_y)}, \overset{6}{help(v_y)}, \overset{7}{help(u_y)})$$

of y and the label D_x of x).

We assert that the same labels R_x and R_y suffice for a routing decision in case $x \in \partial A$. The neighbors u_x and v_x of x are either both vertices of ∂B, or one of them belong to ∂A and another to ∂B. In the second case only the distance from y to the vertex of ∂A can be decoded using R_x and R_y, say $d(u_x, y)$. If $d(x,y) = d(u_x, y) + 1$, then the message is forwarded to u_x, otherwise it is sent to v_x. Finally, if $u_x, v_x \in \partial B$, then the routing decision can be taken by employing the items $d(b, q'), d(b, q'')$ of D_y (here q' and q'' are the end vertices of $Pr(y, Z(A,B))$).

function routing_decision(R_x, R_y)

if $R_x(6) \neq 1$ then

if distance_graphs$(R_x(1), R_y(1))$ = distance_graphs$(R_x(2), R_y(1)) + 1$ then output $R_x(4)$

else output $R_x(5)$

else if $R_x(7) \neq 1$ then

if distance_graphs$(R_x(1), R_y(1))$ = distance_graphs$(R_x(3), R_y(1)) + 1$ then output $R_x(5)$

else output $R_x(4)$

else extract $D_y(4)$ from $R_y(1)$

extract $D_{v_x}(3)$ from $R_x(2)$
extract $D_{u_x}(3)$ from $R_x(3)$
if $D_{u_x}(3) \le D_{v_x}(3)$ then
    if $D_y(4) \le D_{u_x}(3)$ then output $R_x(5)$ else output $R_x(4)$
else
    if $D_y(4) \le D_{v_x}(3)$ then output $R_x(4)$ else output $R_x(5)$

4.4 Routing decision

Here we explain how, using the decomposition tree $T(G)$, one can route between any two vertices of G. The method is very similar to the one we used for distance decoding.

Let again v be a median vertex of G and $u_0, \dots, u_{k-1}$ be its neighbors in counterclockwise order around v. For any vertex $x \in C_v(u_i)$ and index $j = i, i+1, i+2 \ (mod\ k)$, denote by R_x^j the routing label of x with respect to the cut $\{A'_j, B'_j\}$ (R_x was defined in the previous subsection with respect to an arbitrary cut $\{A, B\}$). Let $S(x)$ be the deepest node of the decomposition tree $T(G)$ of G containing x and A_x be the label of $S(x)$ in the NCA-depth labeling scheme of $T(G)$. Let also $S_0, S_1, \dots, S_h$ be the nodes of the path of $T(G)$ from the root (G, v) (which is S_0) to the node $S(x) = S_h$. Denote as before by G_i a subgraph of G induced by $C_v(u_i)$ ($i \in \{0, \dots, k-1\}$).

In the routing labeling scheme, the label $L(x)$ will be the concatenation of A_x, and $h+1$ tuples $\mu_0^x, \mu_1^x, \dots, \mu_h^x$ where μ_q^x ($q \in \{0, \dots, h\}$) is defined as follows. Let S_q be a node (G_q, v_q) of $T(G)$. Assume that x belongs to a cone $C_{v_q}(u_i)$ of G_q for some $i \in \{0, ..., \delta_{G_q}(v_q) - 1\}$. Then,

$$\mu_q^x := (\overset{1}{i},\ \overset{2}{\delta_{G_q}(v_q)},\ \overset{3}{port_{G_q}(x, v_q)},\ \overset{4}{R_x^i},\ \overset{5}{R_x^{i+1}},\ \overset{6}{R_x^{i+2}},\ \overset{7}{port_{G_q}(v_q, x)}).$$

If $x = v_q$, we set $\mu_q^{v_q} := (\delta_{G_q}(v_q), \delta_{G_q}(v_q), 0, 0, 0, 0, 0)$.

Clearly, again $L(x)$ is of length $O(\log^2 n)$ bits for any $x \in V$ and computation of those tuples can be incorporated into the algorithm for building $T(G)$, leading to an $O(n \log n)$ time computation of all labels $L(x), x \in V$ (for a vertex x of a graph G_q, the special neighbors v_x^j and u_x^j can be computed by running Bread-First-Searches from $Z(A'_j, B'_j)$ ($j = i, i+1, i+2$)).

Algorithm ROUTING_DECISION: Routing decision for trigraphs.

Input: two labels $L(x) = A_x \circ \mu_0^x \circ \mu_1^x \circ \cdots \circ \mu_h^x$ and $L(y) = A_y \circ \mu_0^y \circ \mu_1^y \circ \cdots \circ \mu_q^y$.
Output: $port_G(x, y)$, the port number of the first edge on a shortest path from x to y in G.
Method:
use A_x and A_y to find the depth l in $T(G)$ of
    the nearest common ancestor of $S(x)$ and $S(y)$;

extract from $L(x)$ and $L(y)$ the tuples μ_l^x and μ_l^y;
if $\mu_l^x(1) = \mu_l^x(2)$ then output $\mu_l^y(7)$ and stop; /* $x = v_q$ */
if $\mu_l^y(1) = \mu_l^y(2)$ then output $\mu_l^x(3)$ and stop; /* $y = v_q$ */
/* if the cones are 1-neighboring */
if ($\mu_l^x(1) = \mu_l^y(1) - 1$ or $\mu_l^y(1) = 0$ and $\mu_l^x(1) = \mu_l^x(2) - 1$) then output
routing_decision($\mu_l^x(5), \mu_l^y(4)$) and stop;
if ($\mu_l^y(1) = \mu_l^x(1) - 1$ or $\mu_l^x(1) = 0$ and $\mu_l^y(1) = \mu_l^x(2) - 1$) then output
routing_decision($\mu_l^x(4), \mu_l^y(5)$) and stop;
/* if the cones are 2-neighboring */
if ($\mu_l^x(1) = \mu_l^y(1) - 2$ or $\mu_l^y(1) = 0$ and $\mu_l^x(1) = \mu_l^x(2) - 2$ or
$\mu_l^y(1) = 1$ and $\mu_l^x(1) = \mu_l^x(2) - 1$) then output
routing_decision($\mu_l^x(6), \mu_l^y(4)$) and stop;
if ($\mu_l^y(1) = \mu_l^x(1) - 2$ or $\mu_l^x(1) = 0$ and $\mu_l^y(1) = \mu_l^x(2) - 2$ or
$\mu_l^x(1) = 1$ and $\mu_l^y(1) = \mu_l^x(2) - 1$) then output
routing_decision($\mu_l^x(4), \mu_l^y(6)$) and stop;
else output $\mu_l^x(3)$.

Thus, we have

Theorem. *The family of n-vertex trigraphs admits distance and routing labeling schemes with labels of size $O(\log^2 n)$ bits per vertex and constant time distance and routing decisions. Moreover, the schemes are constructable in $O(n \log n)$ time.*

5 Conclusion

In this paper, we considered a class of cellular networks, called irregular cellular networks, which allow a non-uniform distribution of base stations and a non-uniform cell size. The communication graph of such a cellular network forms a trigraph, i.e., a plane triangulation with inner vertices of degree at least six. We have shown that each trigraph with n vertices admits a labeling that assigns $O(\log^2 n)$ bit labels to vertices of the graph such that the distance between any two vertices u and v can be determined in constant time by merely inspecting the labels of u and v, without using any other information about the graph. Furthermore, we have shown that there is a labeling, assigning labels of size $O(\log^2 n)$ bits to vertices, which allows, given the label of a source vertex and the label of a destination, to compute in constant time the port number of the edge from the source that heads in the direction of the destination. These two results for trigraphs provide elegant solutions to a few problems in irregular cellular networks. The distance labeling scheme allows efficient implementation of the distance-based tracking protocol, by providing information, generally not available to the user, and means for accurate cell distance determination. The routing and distance labeling schemes provide compact and efficient routing and connection re-routing protocols.

We conclude this paper with two related open problems not addressed in this work.

Deployment Problem: Given a service area R with obstacles and demands (different subregions may have different demands), deploy minimum number of BSs (perhaps, re-using the maximum number of old BSs) such that the deployed BSs cover entire region R (satisfying all demands) and their communication graph is a trigraph.

Channel Assignment Problem: Investigate the problem of optimal $L(p_1, p_2, \ldots, p_k)$-coloring [4] in trigraphs.

Our future plans are to address these issues.

References

1. I.F. Akyildiz, J.S.M. Ho, and Y.B. Lin, Movement-based location update and selective paging for PCS networks, *IEEE/ACM Trans. Networking,* 4 (1996), 629-638.
2. H.-J. Bandelt, V. Chepoi, Decomposition and l_1-embedding of weakly median graphs, *Europ. J. Combin.*, 21 (2000), 701–714.
3. A. Bar-Noy, I. Kessler, and M. Sidi, Mobile users: to update or not to update, *Wireless Networks,* 1 (1994), 175-185.
4. A.A. Bertossi, M.C. Pinotti, R.B. Tan, Channel assignment with separation for interference avoidance in wireless networks, *IEEE Transactions on Parallel and Distributed Systems*, 14 (2003), 222-235.
5. U. Black, *Mobile and Wireless Networks*, Prentice Hall, 1996.
6. V. Chepoi, F.F. Dragan, Y. Vaxès, Center and diameter problems in plane triangulations and quadrangulations, in *Proc. 13th Ann. ACM-SIAM Symp. Discrete Algorithms (SODA 2002)*, ACM–SIAM 2002, pp. 346-355.
7. V. Chepoi, F.F. Dragan, and Y. Vaxès, Addressing, distances and routing in triangular systems with applications in cellular and sensor networks, in *18th International Parallel and Distributed Processing Symposium (IPDPS 2004)*, CD-ROM / Santa Fe, New Mexico, USA, *IEEE Computer Society*, 2004.
8. V. Chepoi, F.F. Dragan, and Y. Vaxès, Distance and routing labeling schemes for non-positively curved plane graphs, *Technical report 13-2003*, Laboratoire d'Informatique Fondamentale de Marseille, http://www.lim.univ-mrs.fr/Rapports/13-2003-Chepoi-Dragan-Vaxes.pdf.
9. V. Chepoi, C. Fanciullini, Y. Vaxès, Median problem in some plane triangulations and quadrangulations, *Computational Geometry*, 27 (2004), 193–210.
10. G. Fan and J. Zhang, Virtual cellular networks for non-uniformly distributed base stations, in *Proceedings of the 30th Annual International Conference on Parallel Processing*, Valencia, Spain, September 2001.
11. G. Fan and J. Zhang, A novel geometric diagram and its application in wireless networks, *INFOCOM* 2004, IEEE.
12. D. Harel, R. Tarjan, Fast algorithms for finding nearest common ancestor, *SIAM J. Comput.,* 13 (1984), 338–355.
13. J. Li, H. Kameda, and K. Li, Optimal dynamic location update for PCS networks, *IEEE/ACM Trans. Networking,* 8 (2000), 319-327.
14. F.G. Nocetti, I. Stojmenovic, and J. Zhang, Addressing and routing in hexagonal network with application for tracking mobile users and connection rerouting in cellular networks, *IEEE Trans. on Parallel and Distributed Systems,* 13 (2002), 963-971.
15. D. Peleg, Informative labeling schemes for graphs, in *Proceedings of the 25th International Symposium "Mathematical Foundations of Computer Science" (MFCS 2000)*, Bratislava, Slovakia, August 28 - September 1, 2000, *Lecture Notes in Computer Science 1893*, Springer, pp. 579-588, 2000.
16. F.P. Preparata and M.I. Shamos, *Computational Geometry - An Introduction*, Springer-Verlag, 1985.
17. T.S. Rappaport, *Wireless Communications - Principles and Practice*, Prentice Hall, 1996.
18. D.B. West, *Introduction to Graph Theory*, Prentice Hall, 1996.