Proximity Graphs inside Large Weighted Graphs

Bernardo M. Ábrego* Ruy Fabila-Monroy† Silvia Fernández-Merchant*
David Flores-Peñaloza‡ Ferran Hurtado§ Henk Meijer¶ Vera Sacristán§
Maria Saumell§

Abstract

Given a weighted graph $G = (V, E)$ and a subset U of V, we define several graphs with vertex set U in which two vertices are adjacent if they satisfy a specific proximity rule. These rules use the shortest path distance in G and generalize the proximity rules that generate some of the most common proximity graphs in Euclidean spaces. We prove basic properties of the defined graphs and provide algorithms for their computation.

Keywords Neighborhood; proximity graphs; Voronoi diagrams; weighted graphs.

1 Introduction

Consider a graph $G = (V, E)$ whose edges have a positive associated weight, in which the focus of a study is on a subset U of the nodes. Suppose that we have to decide which elements of U are close to other elements of U within the graph, or we must find a suitable subgraph that spans the elements in U, or we have to describe the relative positions in the network of the elements of U. This is nothing else than describing the *proximity relations* of U as a substructure of the network.

A logical option to be carefully explored in this setting is to adapt or mimic geometric proximity graphs, which have been successful in metric scenarios, particularly the Delaunay family of graphs. This is the scope of this paper. To provide notions of closeness, we use generalizations of the nearest neighbor graph, the minimum spanning tree, the relative neighborhood graph, the Gabriel graph, and the Delaunay graph. These generalizations use the shortest path distance along the edges of G. We systematically study the relations among these graphs, their computation in terms of the nodes and the network itself, and the fundamental variations that

*Department of Mathematics, California State University, Northridge, CA, USA ({bernardo.abrego,silvia.fernandez}@csun.edu).

†Departamento de Matemáticas, CINVESTAV, Mexico DF, Mexico (ruyfabila@math.cinvestav.edu.mx).

‡Instituto de Matemáticas, Universidad Nacional Autónoma de México (dflores@math.unam.mx).

§Deptartament de Matemàtica Aplicada II, Universitat Politècnica de Catalunya, Barcelona, Spain ({ferran.hurtado,vera.sacristan,maria.saumell}@upc.edu). Partially supported by projects MTM2009-07242 and Gen. Cat. DGR 2009SGR1040.

¶Science Department, Roosevelt Academy, Middelburg, The Netherlands (h.meijer@roac.nl).

arise when only the vertices of the input graph are taken as generators for the influence regions, or when arbitrary points on the edges may also play this role.

Motivation. Nowadays, many complex relation systems are represented as large networks. In some cases, as in transportation networks or circuit layouts, they correspond to physical situations in which a geometric framework is inherent or implicit. In others, the relationship is essentially combinatorial; important representatives of this situation include the world-wide web, social networks, commercial networks, and citation networks. Whichever the scenario, handling these usually huge data sets has become crucial and requires combined efforts from different fields, including in particular discrete mathematics and computer science [3].

Identifying communities, contrasting local characteristics with global properties within a network, or developing data mining methods are among the basic tasks in network analysis. In this context, measuring proximity among nodes is a key tool. Defining suitable notions of closeness among vertices of a graph help to achieve several goals such as comparing similarity, or finding clusters, spanning structures, or tools allowing good interpolation. The proximity structures we define might also provide a way to simplify the original network and obtain new graphs that retain some of its information.

Related work. Some variations of the problem we study have been considered before in the literature. One of these variations is the case when the graph is purely combinatorial and the distance between two vertices is the minimum number of edges of any path connecting them [1, 11]. However, we find that this approach does not completely provide rich analysis tools. Another related direction of research considers proximity inside an embedded geometric graph while using regions of interaction that are defined in the full Euclidean plane [10, 14, 22].

A different situation arises when the network in embedded in a metric space. Here one may consider the Voronoi diagram inside the network, focusing just on the nodes and the connections or extending the implications of the resulting partition of the graph to the surrounding space. These situations nicely fit phenomena occurring alongside a network, or in its area of influence, and have been intensively investigated as *network Voronoi diagrams* (see [17] and [18] for a thorough description and multiple references) or from the viewpoint of *time metrics* [2, 19, 4]. If the surrounding space is ignored and the focus is on a geometric or weighted graph, the Voronoi partition is still a powerful analysis tool that has been repeatedly studied, as in [16, 7, 12].

Let us finally mention that the set U together with the shortest-path distance on G constitutes a finite metric space. Therefore, our problem may be seen as a particular case of proximity graphs defined on general metric spaces. However, to the best of our knowledge, the precise topic of our work has not been systematically investigated and only a couple of definitions and easily-derived relations among the proximity graphs have been established (see Section 4.5 in [23], and also [13]).

2 Definitions and Notation

We deal with a pair of a connected and edge-weighted graph $G = (V, E)$ and a subset $U \subseteq V$. For simplicity, we write $G = (V, U, E)$ and we define $|V| = m$, $|U| = n$, and $|E| = e$. The edge set E can be considered as a set of interior disjoint, continuous curves in a suitable ambient space. Each edge $e = (v_1, v_2)$ with (positive) weight $w(e)$ is isometric to the interval $[0, w(e)]$, and its endpoints are v_1 and v_2. The shortest path distance together with the edge set constitutes a metric space, and we also refer to this metric space as G. When we speak of a *point of* G we refer to either an endpoint or an interior point of an edge. The distance between a pair of vertices of G as a metric space corresponds to the distance in G as a weighted graph, with the gained advantage that we may extend this definition to points in the interior of edges. The *distance* $d_G(p, q)$ between two points p and q of G is thus the minimum over the weights of all paths connecting p and q in G. The *closed disk* $D_G(p, r)$ is defined as the set of points q of G for which $d_G(p, q) \leq r$. If $i \neq j$, we say that $u_i \in U$ is a *nearest neighbor* of $u_j \in U$ if $d_G(u_j, u_i) \leq d_G(u_j, u_k)$ for all vertices $u_k \in U$ different from u_j. A *midpoint* of two points p and q of G is a point m on one of the shortest paths from p to q such that $d_G(m, p) = d_G(m, q)$. We denote the set of midpoints of p and q by $M_G(p, q)$.

When using empty regions, such as disks, as proximity criteria in G two main variations arise, since we might allow these disks to be centered at any point of G, or restrict their centers to lie only on vertices of the graph, as in [11, 1]. Moreover, the definition of certain regions of interference such as the Gabriel disk might depend on the multiplicity of paths or distances in G. Degeneracies that occur in the standard geometric case, such as non-uniqueness of the nearest neighbor, also generate several possibilities. For the sake of clarity we first present the situation where there are essentially no degeneracies (Sections 3–5). In Section 6 we drop the non-degeneracy assumptions and extend our results to the general setting.

More precisely, in Sections 3, 4, and 5 we consider the case where the following non-degeneracy assumptions are satisfied:

(USP) Unique Shortest Path: for all $u_i, u_j \in U$, the shortest path connecting u_i and u_j is unique;

(NIT) No Isosceles Triangles: there do not exist three distinct vertices $u_i, u_j \in U$, $v \in V - U$ such that $d_G(v, u_i) = d_G(v, u_j)$;

(NRD) No Repeated Distances: there do not exist vertices $v_i, v_j \in V$, $u_i, u_j \in U$ such that $d_G(v_i, u_i) = d_G(v_j, u_j)$ and $v_i \neq u_i$;

(DLP) Different Length Paths: all paths in G connecting distinct nodes in V have different lengths.

Obviously, the previous assumptions are not independent (DLP implies USP, NIT, and NRD; NRD implies NIT), but considering them separately allows us to clarify and provide a more precise description of the scenario. In Section 6, we extend the results from Sections 3–5 to the general case where USP, NIT, NRD, and DLP are not necessarily satisfied.

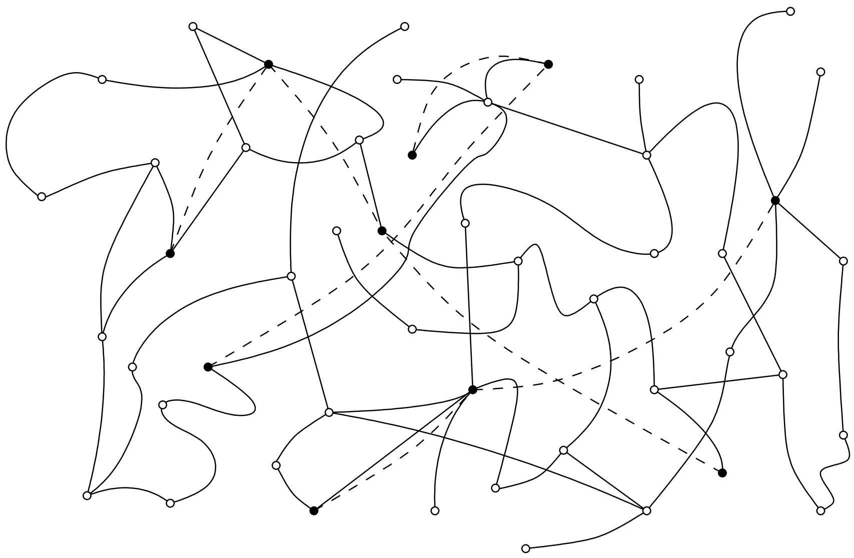

Figure 1: A graph G and the nearest neighbor graph of U in G. The black vertices belong to U, while the white vertices belong to $V - U$. The edges of G are solid and the edges of $\mathrm{NNG}(U;G)$ are dashed. The weight of the edges of G corresponds to their length.

We now adapt several known definitions to proximity structures in graphs $G = (V, U, E)$.

Definition 2.1. The *nearest neighbor graph* of U in $G = (V, U, E)$, denoted by $\mathrm{NNG}(U;G)$, is the graph $H = (U, F)$ such that $(u_i, u_j) \in F$ if and only if u_j is one of the nearest neighbors of u_i in G. See Figure 1 for an example.

Definition 2.2. A *minimum spanning tree* of U in $G = (V, U, E)$ is a tree $T = (U, F)$ such that the sum of $d_G(u_i, u_j)$ over all edges $(u_i, u_j) \in F$ is minimal. The *union of the minimum spanning trees* of U in G, denoted by $\mathrm{UMST}(U;G)$, is the graph consisting of all the edges included in any of the minimum spanning trees of U in G.

If NRD holds, all distances between vertices in U are different. This in particular implies that each vertex in U has exactly one nearest neighbor and that the minimum spanning tree of U in G, denoted by $\mathrm{MST}(U;G)$, is unique.

Definition 2.3. The *relative neighborhood graph* of U in $G = (V, U, E)$, denoted by $\mathrm{RNG}(U;G)$, is the graph $H = (U, F)$ such that $(u_i, u_j) \in F$ if and only if there exists no vertex $u_k \in U$ such that $d_G(u_k, u_i) < d_G(u_i, u_j)$ and $d_G(u_k, u_j) < d_G(u_i, u_j)$.

We propose two basic ways of generalizing the Gabriel graph and the Delaunay graph. One of them (the *free* versions) models continuous situations and allows to center the disks at any point of G, while the other (the *constrained* versions) models discrete situations and requires that disks are centered at vertices of G.

Definition 2.4. The *free-one Gabriel graph* of U in $G = (V, U, E)$, denoted by $\mathrm{GG}_{\mathrm{f1}}(U;G)$, is the graph $H = (U, F)$ such that $(u_i, u_j) \in F$ if and only if there exists a point $p \in M_G(u_i, u_j)$ such that no vertex $u_k \in U$ $(u_k \neq u_i, u_j)$ satisfies $d_G(p, u_k) \leq d_G(p, u_i)$.

Definition 2.5. The *free-all Gabriel graph* of U in $G = (V, U, E)$, denoted by $\mathrm{GG_{fa}}(U; G)$, is the graph $H = (U, F)$ such that $(u_i, u_j) \in F$ if and only if, for each $p \in M_G(u_i, u_j)$, no vertex $u_k \in U$ $(u_k \neq u_i, u_j)$ satisfies $d_G(p, u_k) \leq d_G(p, u_i)$.

If USP holds, the two definitions coincide and we denote the graph by $\mathrm{GG_f}(U; G)$.

Definition 2.6. The *constrained-one Gabriel graph* of U in $G = (V, U, E)$, denoted by $\mathrm{GG_{c1}}(U; G)$, is the graph $H = (U, F)$ such that $(u_i, u_j) \in F$ if and only if there exists a closed disk $D_G(v, r)$ enclosing u_i and u_j and no other vertex from U, where $v \in V$ and $r = \min_{w \in V}\{\rho \mid D_G(w, \rho)$ contains both u_i and $u_j\}$.

Definition 2.7. The *constrained-all Gabriel graph* of U in $G = (V, U, E)$, denoted by $\mathrm{GG_{ca}}(U; G)$, is the graph $H = (U, F)$ such that $(u_i, u_j) \in F$ if and only if every closed disk $D_G(v, r)$ enclosing u_i and u_j does not contain any other vertex of U, where $v \in V$ and $r = \min_{w \in V}\{\rho \mid D_G(w, \rho)$ contains both u_i and $u_j\}$.

If NRD holds, the two definitions coincide and we denote the graph by $\mathrm{GG_c}(U; G)$.

Definition 2.8. The *Voronoi region* of a vertex $u_i \in U$ is the set of points p of G such that $d_G(p, u_i) \leq d_G(p, u_j)$ for all vertices $u_j \in U$ different from u_i. The *Voronoi diagram* of U in $G = (V, U, E)$, denoted by $\mathrm{VD}(U; G)$, is the decomposition of G into the Voronoi regions of the vertices of U.

Definition 2.9. The *free Delaunay graph* of U in $G = (V, U, E)$, denoted by $\mathrm{DG_f}(U; G)$, is the graph $H = (U, F)$ such that $(u_i, u_j) \in F$ if and only if there exists a closed disk $D_G(p, r)$, where p is a point of G, containing u_i and u_j and no other vertex from U.

Definition 2.10. The *constrained Delaunay graph* of U in $G = (V, U, E)$, denoted by $\mathrm{DG_c}(U; G)$, is the graph $H = (U, F)$ such that $(u_i, u_j) \in F$ if and only if there exists a closed disk $D_G(v, r)$, with $v \in V$, containing u_i and u_j and no other vertex from U.

3 Inclusion Sequence

As in the case of the corresponding proximity graphs in Euclidean spaces, the graphs just defined satisfy some inclusion relations. In this section we show which proximity graphs are subgraphs of which other proximity graphs assuming USP, NIT, and NRD.

Lemma 3.1. *For each graph* $G = (V, U, E)$ *we have* $\mathrm{NNG}(U; G) \subseteq \mathrm{MST}(U; G) \subseteq \mathrm{RNG}(U; G) \subseteq \mathrm{GG_f}(U; G) \subseteq \mathrm{DG_f}(U; G)$.

Proof. The proofs are analogous to those for proximity graphs in Euclidean spaces and, in some cases, the more general theory of proximity graphs defined in metric spaces applies (see Section 4.5 in [23]). It should be noticed that the inclusion $\mathrm{RNG}(U; G) \subseteq \mathrm{GG_f}(U; G)$ relies on the assumption of NIT (see Theorem 6.1). ⊠

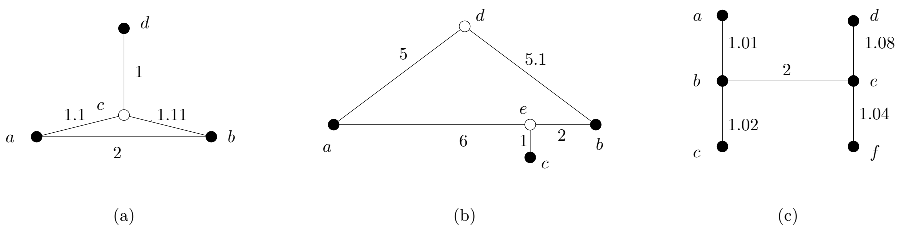

Figure 2: Example graphs; the black vertices belong to U, while the white vertices belong to $V - U$.

Lemma 3.2. *For each graph $G = (V, U, E)$ we have* $\mathrm{NNG}(U; G) \subseteq \mathrm{DG_c}(U; G)$.

Proof. Let (u_i, u_j) be an edge of $\mathrm{NNG}(U; G)$, where u_j is the nearest neighbor of u_i. The closed disk $D_G(u_i, r)$, where $r = d_G(u_i, u_j)$, contains no vertices from U different from u_i, u_j. Thus $(u_i, u_j) \in \mathrm{DG_c}(U; G)$. ⊠

Lemma 3.3. *For each graph $G = (V, U, E)$ we have* $\mathrm{GG_c}(U; G) \subseteq \mathrm{DG_c}(U; G) \subseteq \mathrm{DG_f}(U; G)$.

Proof. It follows from the definitions of $\mathrm{GG_c}(U; G)$, $\mathrm{DG_c}(U; G)$, and $\mathrm{DG_f}(U; G)$. ⊠

Lemma 3.4. *There exist graphs $G_1 = (V_1, U_1, E_1)$, $G_2 = (V_2, U_2, E_2)$, and $G_3 = (V_3, U_3, E_3)$ for which* $\mathrm{NNG}(U_1; G_1) \not\subset \mathrm{GG_c}(U_1; G_1)$, $\mathrm{GG_c}(U_2; G_2) \not\subset \mathrm{GG_f}(U_2; G_2)$, *and* $\mathrm{MST}(U_3; G_3) \not\subset \mathrm{DG_c}(U_3; G_3)$.

Proof. Let G_1 be the graph on Figure 2a. The graph $\mathrm{NNG}(U_1; G_1)$ contains the edges (a, b) and (a, d) while $\mathrm{GG_c}(U_1; G_1)$ only contains the edge (a, d). This proves that there are cases in which $\mathrm{NNG}(U; G) \not\subset \mathrm{GG_c}(U; G)$. To see that $\mathrm{GG_c}(U; G) \subseteq \mathrm{GG_f}(U; G)$ does not hold in general, let G_2 be the graph on Figure 2b. We have that $\mathrm{GG_c}(U_2; G_2)$ contains the edge (a, b) whereas $(a, b) \notin \mathrm{GG_f}(U_2; G_2)$. Finally, if G_3 is the graph on Figure 2c, $\mathrm{MST}(U_3; G_3)$ contains the edge (b, e) but $\mathrm{DG_c}(U_3; G_3)$ does not. Thus there exist graphs $G = (V, U, E)$ such that $\mathrm{MST}(U; G) \not\subset \mathrm{DG_c}(U; G)$. ⊠

These lemmas are sufficient to prove the following theorem:

Theorem 3.5. *The inclusion relations among all classes of proximity graphs are shown in Table 1. The symbol $\subset$ means that the inclusion is satisfied for all graphs G, and $\not\subset$ means that there are graphs G for which the inclusion is not satisfied.*

It is not difficult to produce examples proving that all inclusions in the table are proper, in the sense that there exist graphs G for which the corresponding proximity subgraph does not coincide with its supergraph.

Table 1: Inclusion relations among proximity graphs assuming USP, NIT, and NRD.

	MST	RNG	$\mathrm{GG_c}$	$\mathrm{GG_f}$	$\mathrm{DG_c}$	$\mathrm{DG_f}$
NNG	$\subset$	$\subset$	$\not\subset$	$\subset$	$\subset$	$\subset$
MST		$\subset$	$\not\subset$	$\subset$	$\not\subset$	$\subset$
RNG			$\not\subset$	$\subset$	$\not\subset$	$\subset$
$\mathrm{GG_c}$				$\not\subset$	$\subset$	$\subset$
$\mathrm{GG_f}$					$\not\subset$	$\subset$
$\mathrm{DG_c}$						$\subset$

4 Geometric and Combinatorial Properties

In this section we study geometric and combinatorial properties of the previously defined proximity graphs. We start by proving that the free Delaunay graph is the dual graph of the Voronoi diagram, and by showing the consequences of this result on the combinatorial complexity of the graphs under study. Then we deal with two important particular cases, namely, when G is planar and when G is a tree. Finally, we characterize the graphs that are isomorphic to a proximity graph of another graph.

Remark 4.1. Consider $G = (V, U, E)$ and assume that NIT is satisfied. If a point p is in the intersection of the Voronoi regions of two vertices u_i and u_j, then p is a point on an edge of G. Moreover, one of the endpoints of this edge belongs to the Voronoi region of u_i and the other one belongs to the Voronoi region of u_j. So if q is a point in the intersection of a different pair of Voronoi regions, then p and q are on different edges of G. In particular, the intersection of three Voronoi regions is empty.

We define the *dual graph of the Voronoi diagram* of $G = (V, U, E)$ as the graph with vertex set U and edges connecting two vertices if and only if their Voronoi regions share a point of G that does not belong to the Voronoi region of any other vertex in U.

Proposition 4.2. *Let* $G = (V, U, E)$ *be a graph. Then* $\mathrm{DG_f}(U; G)$ *is the dual graph of* $\mathrm{VD}(U; G)$.

Proof. If the Voronoi regions of two vertices $u_i \neq u_j \in U$ intersect in a point y that is not in the Voronoi region of any other vertex in U, then the closed disk $D_G(y, r)$, where r is the distance from y to u_i, contains u_i, u_j, and no other vertex from U. Thus (u_i, u_j) is an edge of $\mathrm{DG_f}(U; G)$.

Inversely, let us assume that there exists a disk $D_G(x, r)$, where x is a point of G, containing only two vertices of U, u_i and u_j. If $d_G(x, u_i) = d_G(x, u_j)$, then x lies in the Voronoi regions of u_i and u_j (and does not lie in the Voronoi region of any other vertex from U). Otherwise assume that $d_G(x, u_j) > d_G(x, u_i)$. Since $d_G(u_j, u_j) < d_G(u_j, u_i)$ and the distance in G is a continuous function, there exists a point x' in the shortest path from x to u_j such that $d_G(x', u_j) = d_G(x', u_i)$. If we define $r' = d_G(x', u_j)$, we have that $D_G(x', r')$ is contained in

$D_G(x, r)$ and has u_i and u_j on its boundary. Hence x' lies in the Voronoi regions of u_i and u_j (and does not lie in the Voronoi region of any other vertex from U). ⊠

The previous proposition is a key tool to prove other results of interest, such as the following:

Corollary 4.3. *Let $G = (V, U, E)$ be a graph. The number of edges of* $\mathrm{NNG}(U; G)$, $\mathrm{MST}(U; G)$, $\mathrm{RNG}(U; G)$, $\mathrm{GG_c}(U; G)$, $\mathrm{GG_f}(U; G)$, $\mathrm{DG_c}(U; G)$, *and* $\mathrm{DG_f}(U; G)$ *is at most e.*

Proof. Let (u_i, u_j) be an edge of $\mathrm{DG_f}(U; G)$. By Proposition 4.2, the Voronoi regions of u_i and u_j intersect at a point p of G such that $d_G(p, u_i) = d_G(p, u_j) < d_G(p, u_k)$ for all vertices $u_k \in U$ different from u_i and u_j. By Remark 4.1, p is a point on an edge of G and this edge does not include any other point on the intersection of the Voronoi regions of two vertices in U. The number of edges of $\mathrm{DG_f}(U; G)$ is therefore less than or equal to e.

Since, for all other classes, the proximity graph on G is a subgraph of $\mathrm{DG_f}(U; G)$, their size is also bounded by e. ⊠

The next proposition shows that this bound is tight for $\mathrm{RNG}(U; G)$, $\mathrm{GG_f}(U; G)$, and $\mathrm{DG_f}(U; G)$, and is tight up to a constant factor for $\mathrm{GG_c}(U; G)$ and $\mathrm{DG_c}(U; G)$.

Proposition 4.4. *There exists a graph $G = (V, U, E)$ such that* $\mathrm{RNG}(U; G) = \mathrm{GG_f}(U; G) = \mathrm{DG_f}(U; G) = G$. *There also exists a graph $G' = (V', U', E')$ such that the number of edges of* $\mathrm{GG_c}(U'; G')$ *and* $\mathrm{DG_c}(U'; G')$ *is $e'/2$. Furthermore, all of these proximity graphs have $\Theta(n^2)$ edges.*

Proof. Consider two groups of $n/2$ vertices. Let U be this set of n vertices, $V = U$ and $G = (V, U, E)$ be the complete bipartite graph on the two groups of vertices. The weights of the edges of G are real numbers between 1 and 1.5, and can be assigned so that G satisfies DLP (and thus USP, NIT, and NRD as well). In this situation $\mathrm{RNG}(U; G) = \mathrm{GG_f}(U; G) = \mathrm{DG_f}(U; G) = G$, and $\mathrm{RNG}(U; G)$, $\mathrm{GG_f}(U; G)$, and $\mathrm{DG_f}(U; G)$ have $n^2/4$ edges.

Now add vertices to G in such a way that there exists a vertex in $V - U$ at the midpoint of each edge, and break each edge coherently. Perturb these new vertices so that the graph still fulfills DLP. Let $G' = (V', U', E')$ be the resulting graph. In this case the number of edges of $\mathrm{GG_c}(U'; G')$ and $\mathrm{DG_c}(U'; G')$ is $e'/2 = n^2/4$. ⊠

We now consider the case where $G = (V, U, E)$ is a planar graph.

Theorem 4.5. *Let $G = (V, U, E)$ be a planar graph. Then* $\mathrm{DG_f}(U; G)$ *is planar.*

Proof. Consider a plane drawing of G. Let (u_i, u_j) be an edge of $\mathrm{DG_f}(U; G)$. By Proposition 4.2, there exists a point p of G (not in V by Remark 4.1) such that $d_G(p, u_i) = d_G(p, u_j) < d_G(p, u_k)$ for all vertices $u_k \in U$ different from u_i and u_j. Draw the edge (u_i, u_j) by following the union of a shortest path in G from u_i to p and a shortest path from p to u_j. Note that each point $q \neq p$ in a shortest path from u_i to p is such that $d_G(q, u_i) < d_G(q, u_k)$ for all vertices

$u_k \in U$ different from u_i, and the same holds for u_j. Draw all edges in $\mathrm{DG_f}(U;G)$ in this way and suppose the graph has one crossing. Since G is planar, this crossing occurs in a vertex w of $V-U$. Then, w belongs to two shortest paths, one from a point $p_l \in G-V$ to a point $u_l \in U$ and the other from $p_h \in G-V$ to $u_h \in U$. Hence, $d_G(w,u_l) < d_G(w,u_\nu)$ for all vertices $u_\nu \in U$ different from u_l, and $d_G(w,u_h) < d_G(w,u_\nu)$ for all vertices $u_\nu \in U$ different from u_h. This yields a contradiction.

Observe that an edge of G might be used in the drawing of several edges of $\mathrm{DG_f}(U;G)$. It is easy to prove that, in this case, it can be duplicated in such a way that no crossings are created. ⊠

Corollary 4.6. *If $G=(V,U,E)$ is a planar graph, then* $\mathrm{NNG}(U;G)$, $\mathrm{MST}(U;G)$, $\mathrm{RNG}(U;G)$, $\mathrm{GG_c}(U;G)$, $\mathrm{GG_f}(U;G)$, *and* $\mathrm{DG_c}(U;G)$ *are planar.*

We have just proved that the proximity graphs inherit planarity from the original graph. Next we show that they also inherit acyclicity.

Theorem 4.7. *Let $G=(V,U,E)$ be a tree. Then* $\mathrm{DG_f}(U;G) = \mathrm{MST}(U;G)$.

Proof. As $\mathrm{MST}(U;G) \subseteq \mathrm{DG_f}(U;G)$, it suffices to show that $\mathrm{DG_f}$ does not contain any cycle.

Suppose that $\mathrm{DG_f}$ contains a cycle $u_1u_2\dots u_lu_1$ $(3 \le l \le n)$. By Proposition 4.2, the intersection of the Voronoi regions of every pair of points u_i, u_{i+1} is non-empty; let $p_{i,i+1}$ be a point belonging to this intersection. By Remark 4.1, the points $p_{i,i+1}$ are not in V, are pairwise different, and satisfy that $d_G(p_{i,i+1},u_i) = d_G(p_{i,i+1},u_{i+1}) < d_G(p_{i,i+1},u_k)$ for all vertices $u_k \in U$ different from u_i and u_{i+1}. For every vertex u_i, consider the union of the unique path in G from $p_{i-1,i}$ to u_i and the unique path in G from u_i to $p_{i,i+1}$. Let c_i be the unique path in G from $p_{i-1,i}$ to $p_{i,i+1}$. Observe that this path is contained in the previous union. Since the paths from $p_{i-1,i}$ to u_i and from u_i to $p_{i,i+1}$ are also shortest paths, each point q in c_i different from $p_{i-1,i}$ and $p_{i,i+1}$ is such that $d_G(q,u_i) < d_G(q,u_k)$ for all vertices $u_k \in U$ different from u_i. As a consequence, two paths c_i and c_j, $i \ne j$, only intersect if $j = i+1$, and the intersection takes place at point $p_{i,i+1}$. Thus the union of the paths c_i is a cycle of G, contradicting that G is a tree. ⊠

Corollary 4.8. *Let $G=(V,U,E)$ be a tree. Then* $\mathrm{GG_c}(U;G)$ *and* $\mathrm{DG_c}(U;G)$ *are forests, and* $\mathrm{RNG}(U;G) = \mathrm{GG_f}(U;G) = \mathrm{MST}(U;G)$.

Next we give complete characterizations for those graphs that are isomorphic to a certain proximity graph of another graph. Our results are motivated by the rich literature on characterizations for the graphs that are isomorphic to (or can be *drawn as*) one of the usual proximity graphs defined on a point set in the plane (see [15] for an excellent survey).

Proposition 4.9. *If $G=(V,E)$ is a graph, there exists a graph $\bar{G}=(\bar{V},\bar{U},\bar{E})$ such that $G \cong \mathrm{NNG}(\bar{U};\bar{G})$ if and only if G is acyclic and does not contain isolated vertices.*

Proof. On the one hand, any nearest neighbor graph does not contain isolated vertices because every vertex is connected to its nearest neighbor. On the other hand, this proximity graph is acyclic because it is a subgraph of the minimum spanning tree. This settles one of the implications.

Now let us suppose that G does not contain cycles or isolated vertices. We define a new graph $\bar{G} = (\bar{V}, \bar{U}, \bar{E})$ such that $G \cong \mathrm{NNG}(\bar{U}; \bar{G})$. Let $\bar{V} = \bar{U} = V$, $\bar{E} = E$, and look at each connected component of $\bar{G}$ as a rooted tree. Connect all pairs of roots of different trees. Assign weights to the edges of $\bar{G}$ so that the nearest neighbor of each vertex different from a root is its predecessor in its tree, and the nearest neighbor of a root is one of its successors in its tree. The graph $\bar{G}$ satisfies $G \cong \mathrm{NNG}(\bar{U}; \bar{G})$. ⊠

The next characterization is straightforward.

Proposition 4.10. *If $G = (V, E)$ is a graph, there exists a graph $\bar{G} = (\bar{V}, \bar{U}, \bar{E})$ such that $G \cong \mathrm{MST}(\bar{U}, \bar{G})$ if and only if G is a tree.*

Proposition 4.11. *If $G = (V, E)$ is a graph, there exists a graph $\bar{G} = (\bar{V}, \bar{U}, \bar{E})$ such that $G \cong \mathrm{RNG}(\bar{U}; \bar{G})$ if and only if G is triangle-free.*

Proof. Suppose that there exists a graph $\bar{G} = (\bar{V}, \bar{U}, \bar{E})$ such that $G \cong \mathrm{RNG}(\bar{U}; \bar{G})$. Notice that, for every group of three vertices in $\mathrm{RNG}(\bar{U}; \bar{G})$, the edge connecting the furthest pair is not in the graph. Thus $\mathrm{RNG}(\bar{U}; \bar{G})$ is triangle-free and so is G.

Inversely, let $\bar{V} = \bar{U} = V$, $\bar{E} = E$ and assign to all edges in $\bar{E}$ the same weight. Consider two vertices u_i, u_j in $\bar{U}$. If their corresponding vertices in G are adjacent, they are relative neighbors in $\bar{G}$. Indeed, since G is triangle-free, no vertex in G is adjacent to both u_i and u_j, so no vertex in $\bar{U}$ lies in the lens* defined by u_i and u_j in $\bar{G}$. If they are relative neighbors in $\bar{G}$, no other vertex of $\bar{U}$ lies in the shortest path connecting them, and thus their corresponding vertices in G are adjacent. Notice that the graph $\bar{G}$ as described does not satisfy DLP, but this can easily be fixed by applying an infinitesimal perturbation to the weights of its edges. ⊠

Proposition 4.12. *Let $G = (V, E)$ be a graph. There exists a graph $\bar{G} = (\bar{V}, \bar{U}, \bar{E})$ such that $G \cong \mathrm{GG_c}(\bar{U}; \bar{G}) = \mathrm{GG_f}(\bar{U}; \bar{G}) = \mathrm{DG_c}(\bar{U}; \bar{G}) = \mathrm{DG_f}(\bar{U}; \bar{G})$.*

Proof. Let $\bar{U} = V$, $\bar{E} = E$ and assign to all edges in $\bar{E}$ the same weight. Add a new vertex in $\bar{V} - \bar{U}$ at the midpoint of each of the edges in $\bar{E}$, breaking them. Then $G \cong \mathrm{GG_c}(\bar{U}; \bar{G}) = \mathrm{GG_f}(\bar{U}; \bar{G}) = \mathrm{DG_c}(\bar{U}; \bar{G}) = \mathrm{DG_f}(\bar{U}; \bar{G})$. Observe that $\bar{G}$ is degenerate, but the weights of its edges can be slightly modified so that the graph satisfies DLP. ⊠

*The lens defined by two points at distance d has been sometimes called lune in the literature, and it is the intersection of the disks of radius d centered at the points.

5 Algorithms

In this section we provide algorithms to compute each of the proximity graphs we have presented.

Algorithm for $\mathrm{DG}_\mathrm{f}(U;G)$

The Voronoi diagram of U in $G=(V,U,E)$ can be computed in $O(e+(m-n)\log(m-n))$ time using an algorithm proposed in [7]. As shown in Proposition 4.2, $\mathrm{DG}_\mathrm{f}(U;G)$ is its dual graph, so it can be computed scanning the *bridges* of $\mathrm{VD}(U;G)$, that is, the edges with the property that the two endpoints belong to different Voronoi regions. This is done in $O(e)$ extra time.

Algorithm for $\mathrm{DG}_\mathrm{c}(U;G)$

Two different vertices u, $u' \in U$ are adjacent in $\mathrm{DG}_\mathrm{c}(U;G)$ if and only if there exists a closed disk $D_G(v,r)$, with $v \in V$, which is empty of points in U except for u and u'. If such a disk exists, the two vertices in U closest to vertex v are u and u'. For each vertex v_i in V, the algorithm computes its two closest vertices u_i and u'_i of U. Then the edges in $\mathrm{DG}_\mathrm{c}(U;G)$ are (u_i,u'_i) for all i.

To find the two closest vertices in U of every vertex in V we use a technique similar to that in [7, 9, 18]. The sketch of the algorithm is as follows:

1. For each vertex $u \in U$, initialize a tree T_u rooted at u. For each vertex $v \in V$, insert into a min-priority Q two elements v^1, v^2 with labels $d(v^1)$, $d(v^2)$ initially set to infinity (except for the elements u^1, where $u \in U$, which are inserted with value $d(u^1)=0$). The parameters $d(v^1)$ and $d(v^2)$ will respectively store the lengths of the current shortest and second shortest paths from v to any of the vertices in U, with the additional condition that these two shortest paths lead to two different elements of U.

2. Extract the element v^j of Q with smallest label. Add it to $T_{u'}$, where u' is the origin of the path to v^j having length $d(v^j)$. For each neighbor t of v:

   - If t^1 has not yet been extracted from Q, update the labels of t^1 and t^2, with the restriction that the paths with lengths $d(t^1)$ and $d(t^2)$ must lead to two distinct elements of U.
   - If t^1 has been extracted from Q but t^2 has not, update the label $d(t^2)$ if the new path through v^j is shorter and u' is different from the root of the tree containing t^1.

3. Repeat Step 2 until Q is empty.

Our algorithm is very similar to the one presented in [9] to compute the k nearest vertices in U of every vertex in V, which runs in $O(\min\{ne+nm\log m, km\log m+ke\log m\})$ time. So here we omit further details of the steps of the algorithm and the proof of its correctness, which are carefully given in [9], and we concentrate on the implementation details that allow to improve the running time of the algorithm in [9].

Implementation of the Algorithm and Complexity

The min-priority queue Q is implemented via Fibonnacci heaps [8], so that the operations Insert, Find-min, and Decrease-key are done in $O(1)$ amortized time, and the operations Extract-min and Delete are done in $O(\log n)$ time.

For each element v^j of Q, we store the parent $\pi(v^j)$ that v has in the path of length $d(v^j)$, and the element $T(v^j)$ of U that is the origin of this path. We also make use of a table SP of size $m \times 2$ in which we keep track of the number of times an element has been extracted from Q to be added to a tree, and of the trees to which the element has been added. Thus $SP[v, j] = i$ ($j = 1, 2$) means that, the j-th time that vertex v has been extracted from Q, it has been added to T_{u_i}. It is worth pointing out that we do not actually need to store the trees because the only information necessary to compute $\mathrm{DG_c}(U; G)$ is given by table SP.

At each iteration of the algorithm, when processing a neighbor t of a vertex that has been added to a tree, we follow the same steps as in [9], except for the following: If we find a shorter path from t to a vertex in U, we do not delete t^2 and/or t^1 from Q and reinsert them with new sources $T(t^2)$ and/or $T(t^1)$, but we simply decrease the keys of t^2 and/or t^1 (which is done in $O(1)$ amortized time) and update the corresponding values of $\pi(t^2)$, $\pi(t^1)$, $T(t^2)$, and $T(t^1)$.

Let us finally analyze the complexity of the algorithm.

Since the only insertions into Q are done in the initialization step, and at the end of the algorithm Q is empty, the total running time of the insertions and extractions is $O(m \log m)$.

We must also account for the Decrease-key operations. Let v be a vertex of V. The associated vertices v^1, v^2 might have their value in Q decreased only if one of the neighbors of v in G is added to a tree, so at most $2\delta(v)$ times, where $\delta(v)$ is the degree of v in G. This implies that the total number of decrease operations in the algorithm is at most $8e$. Since each of them is done in constant amortized time, the total running time is $O(e)$.

Regarding space, it is clear that the total space used by the algorithm is $O(m + e)$.

Summarizing, we have proved

Theorem 5.1. *For each graph $G = (V, U, E)$ the graph $\mathrm{DG_c}(U; G)$ can be computed in $O(e + m \log m)$ time and $O(e)$ space.*

Algorithm for $\mathrm{GG_f}(U; G)$

Recall that $u_i, u_j \in U$ are adjacent in $\mathrm{GG_f}(U; G)$ if they are the two closest vertices in U of their midpoint. So we propose an algorithm that first computes the shortest path between every pair of vertices in U and afterwards tests the previous condition. To do this last step, we scan the shortest path in $O(m)$ time to find its midpoint and then check whether the midpoint is contained in a bridge between the Voronoi regions of the two vertices in U, which can be done in constant time if the Voronoi diagram of U in G has been precomputed. Observe that, as $\mathrm{GG_f}(U; G) \subseteq \mathrm{DG_f}(U; G)$, it suffices to test the edges in $\mathrm{DG_f}(U; G)$.

To compute all pairs (of vertices in U) shortest paths, we might distinguish two cases, depending on the order of magnitude of the number of edges of G. As far as we know, the best algorithm for the case in which G is a sparse graph has an asymptotic cost of $O(me \log \alpha(m,e))$ in the comparison-addition model (α denotes the functional inverse of Ackermann's function) [21]. In this algorithm, after a preprocessing phase of cost $O(m \log m)$, single-source shortest path queries can be carried out in $O(e \log \alpha(m,e))$ time. Since we are only interested in shortest paths between vertices in U, the total cost of the algorithm in our case is $O(m \log m + ne \log \alpha(m,e))$.

If G is dense, the previous algorithms might be cubic. To the best of our knowledge, given a dense graph $G = (V, E)$, with $|V| = m$ and $|E| = e$, the fastest algorithm to compute the shortest paths between all pairs of vertices in V runs in $O\left(m^3 \log^3 \log m / \log^2 m\right)$ time in the standard RAM model [5].

For practical purposes, we define

$$\mathrm{APSP}(G) = O\left(\min\{m \log m + ne \log \alpha(m,e), m^3 \log^3 \log m / \log^2 m\}\right) .$$

So we have:

Theorem 5.2. *For each graph $G = (V, U, E)$, the graph $\mathrm{GG_f}(U; G)$ can be computed in $O(APSP(G) + \min\{n^2, e\}m)$ time.*

Algorithm for $\mathrm{GG_c}(U; G)$

Given a pair of vertices $u_i, u_j \in U$, there is a straightforward way to know whether (u_i, u_j) is an edge of $\mathrm{GG_c}(U; G)$ if the distances from u_i and u_j to all vertices in V have been precomputed. Firstly, for each $v \in V$, we compute $\max\{d_G(v, u_i), d_G(v, u_j)\}$, which is the radius of the smallest closed disk centered at v containing both u_i and u_j. Secondly, among this family of disks, we pick the one that has the smallest radius. Thirdly, we check whether the chosen disk contains a vertex in U different from u_i and u_j. These three steps can be done in $O(m)$ time.

Since $\mathrm{GG_c}(U; G) \subseteq \mathrm{DG_f}(U; G)$, we only test the edges in $\mathrm{DG_f}(U; G)$. Consequently,

Theorem 5.3. *For each graph $G = (V, U, E)$, the graph $\mathrm{GG_c}(U; G)$ can be computed in $O(APSP(G) + \min\{n^2, e\}m)$ time.*

Algorithm for $\mathrm{RNG}(U; G)$

We propose a two steps algorithm. Firstly, we compute the shortest paths between all pairs of vertices in U to obtain the distances among these vertices. Secondly, we compute $\mathrm{RNG}(U; G)$ by checking, for all pairs of vertices $u_i, u_j \in U$ that are adjacent in $\mathrm{DG_f}(U; G)$, whether there exists any vertex $u_k \in U$ such that $d_G(u_k, u_i) < d_G(u_i, u_j)$ and $d_G(u_k, u_j) < d_G(u_i, u_j)$.

Theorem 5.4. *For each graph $G = (V, U, E)$, the graph $\mathrm{RNG}(U; G)$ can be computed in $O(APSP(G) + \min\{n^2, e\}n)$ time.*

Algorithm for $\mathrm{MST}(U; G)$

A first approach to obtain the minimum spanning tree of U in $G = (V, U, E)$ could be to compute the distances between all pairs of vertices in U and then apply an efficient algorithm to produce the minimum spanning tree of the complete graph on the vertices in U (if any of the supergraphs of $\mathrm{MST}(U;G)$ was already known, some of the edges could be discarded). This algorithm would run in $O(\mathrm{APSP}(G))$ time. Next we show that another analogy between the usual proximity graphs and the new proximity structures on graphs allows to derive a better algorithm.

It is well known that the Gabriel graph of a set of points in the plane contains those edges in the Delaunay graph that intersect their dual Voronoi edges. The analogous property satisfied by our proximity structures on graphs is the following:

Lemma 5.5. *Let $G = (V, U, E)$ be a graph and let (u_i, u_j) be an edge of $\mathrm{DG_f}(U;G)$. The graph $\mathrm{GG_f}(U;G)$ contains (u_i, u_j) if and only if the shortest path between u_i and u_j is contained in the Voronoi regions of u_i and u_j.*

Proof. It suffices to notice that the shortest path between u_i and u_j is contained in the Voronoi regions of u_i and u_j if and only if its midpoint lies in the Voronoi regions of both vertices. ⊠

Remark 5.6. Let $G_1 = (V, E)$, $G_2 = (V, E)$ be two weighted graphs with the same sets of vertices and edges. Let $|e|_1$ (respectively $|e|_2$) denote the weight of edge e in G_1 (respectively G_2). Suppose that $E = E' \cup E''$, where $|e'|_1 = |e'|_2$ for all $e' \in E'$ and $|e''|_1 < |e''|_2$ for all $e'' \in E''$. Suppose also that no edge in E'' belongs to $\mathrm{MST}(V;G_1)$. Then $\mathrm{MST}(V;G_1) = \mathrm{MST}(V;G_2)$.

Now let us consider $\mathrm{DG_f}(U;G)$ as a weighted graph, where the weight of an edge is given by the distance in G between its endpoints. As $\mathrm{MST}(U;G) \subseteq \mathrm{DG_f}(U;G)$, we have that $\mathrm{MST}(U;G) = \mathrm{MST}(U;\mathrm{DG_f}(U;G))$. We define the weighted graph $\widehat{\mathrm{DG}}_\mathrm{f}(U;G)$ as follows. The vertex set and edges are the same as in $\mathrm{DG_f}(U;G)$, and the weight of an edge (u_i, u_j) in $\widehat{\mathrm{DG}}_\mathrm{f}(U;G)$ is the length of the shortest path from u_i to u_j in G that is contained in the Voronoi regions of both vertices. For example, in Figure 2b the shortest path between a and b has length 8, so this is the weight of (a, b) in $\mathrm{DG_f}(U;G)$. However, this path traverses the Voronoi region of c. The shortest path from a to b that is contained in the Voronoi regions of both vertices is the path $\{a, d, b\}$, which has length 10.1, and this is the weight of (a, b) in $\widehat{\mathrm{DG}}_\mathrm{f}(U;G)$.

Observe that the weight of an edge in $\mathrm{DG_f}(U;G)$ is less than or equal to its weight in $\widehat{\mathrm{DG}}_\mathrm{f}(U;G)$, and that, by Lemma 5.5, they are equal if and only if the edge belongs to $\mathrm{GG_f}(U;G)$. Moreover, since $\mathrm{MST}(U;G) \subseteq \mathrm{GG_f}(U;G)$, an edge that has different weights in $\mathrm{DG_f}(U;G)$ and $\widehat{\mathrm{DG}}_\mathrm{f}(U;G)$ does not belong to $\mathrm{MST}(U;G)$. By Remark 5.6, we can conclude that $\mathrm{MST}(U;\widehat{\mathrm{DG}}_\mathrm{f}(U;G)) = \mathrm{MST}(U;\mathrm{DG_f}(U;G))$, so $\mathrm{MST}(U;\widehat{\mathrm{DG}}_\mathrm{f}(U;G)) = \mathrm{MST}(U;G)$.

Our algorithm takes advantage of the fact that computing the weights in $\widehat{\mathrm{DG}}_\mathrm{f}(U;G)$ is easier than in $\mathrm{DG_f}(U;G)$. Firstly, we compute the Voronoi diagram of U in G using the algorithm proposed in [7]. This algorithm provides not only the nearest vertex in U of every vertex in V but also the distance between them. Thus the length of the shortest path between two neighbors u_i, u_j in $\widehat{\mathrm{DG}}_\mathrm{f}(U;G)$ containing a particular bridge of $\mathrm{VD}(U;G)$ can be computed in constant time. In order to obtain the weight of edge (u_i, u_j) in $\widehat{\mathrm{DG}}_\mathrm{f}(U;G)$, all we have to do is to scan all

Table 2: Inclusion relations among all classes of proximity graphs in the general case.

	UMST	RNG	$\mathrm{GG_{ca}}$	$\mathrm{GG_{c1}}$	$\mathrm{GG_{fa}}$	$\mathrm{GG_{f1}}$	$\mathrm{DG_c}$	$\mathrm{DG_f}$
NNG	$\subset$	$\subset$	$\not\subset$	$\not\subset$	$\not\subset$	$\not\subset$	$\not\subset$	$\not\subset$
UMST		$\subset$	$\not\subset$	$\not\subset$	$\not\subset$	$\not\subset$	$\not\subset$	$\not\subset$
RNG			$\not\subset$	$\not\subset$	$\not\subset$	$\not\subset$	$\not\subset$	$\not\subset$
$\mathrm{GG_{ca}}$				$\subset$	$\not\subset$	$\not\subset$	$\subset$	$\subset$
$\mathrm{GG_{c1}}$					$\not\subset$	$\not\subset$	$\subset$	$\subset$
$\mathrm{GG_{fa}}$						$\subset$	$\not\subset$	$\subset$
$\mathrm{GG_{f1}}$							$\not\subset$	$\subset$
$\mathrm{DG_c}$								$\subset$

the bridges between the Voronoi regions of both vertices. All in all, the graph $\widehat{\mathrm{DG}}_{\mathrm{f}}(U;G)$ can be computed in $O(e+(m-n)\log(m-n))$ time.

Afterwards we can apply an efficient algorithm to obtain $\mathrm{MST}(U;\widehat{\mathrm{DG}}_{\mathrm{f}}(U;G))$. Two suitable options are the algorithm in [6], which runs in $O(e\,\alpha(e,n))$ time, or the algorithm in [20], whose exact running time is not known, but which runs in linear time for most graphs.

In conclusion,

Theorem 5.7. *For each graph $G=(V,U,E)$, the graph $\mathrm{MST}(U;G)$ can be computed in $O(e\,\alpha(e,n)+(m-n)\log(m-n))$ time.*

Algorithm for $\mathrm{NNG}(U;G)$

An algorithm to compute the nearest neighbor graph by means of $\mathrm{DG_f}(U;G)$ is also provided in [7]. Its total running time is $O(e+(m-n)\log(m-n))$.

6 Presence of Degeneracies

In this section we generalize our results to the case in which degeneracies arise.

Theorem 6.1. *If degenerate situations are allowed, the inclusion relations among all classes of proximity graphs are shown in Table 2. Furthermore, all classes of proximity graphs are different.*

Proof. First we deal with the new graphs. As far as inclusion relations are concerned, the graph $\mathrm{GG_{c1}}$ behaves essentially as the graph $\mathrm{GG_c}$ in the non-degenerate case, and the same holds for the graphs $\mathrm{GG_{f1}}$ and $\mathrm{GG_f}$. This rule has few exceptions related to NNG, UMST, and RNG which are explained below.

Regarding $\mathrm{GG_{ca}}$, the example in Figure 2a shows that $\mathrm{NNG} \not\subset \mathrm{GG_{ca}}$ and $\mathrm{RNG} \not\subset \mathrm{GG_{ca}}$. Clearly, $\mathrm{GG_{ca}} \subset \mathrm{GG_{c1}}$ and one can easily come up with a situation where $\mathrm{GG_{ca}} \subsetneq \mathrm{GG_{c1}}$. The graph in Figure 2b proves that $\mathrm{GG_{ca}} \not\subset \mathrm{GG_{f1}}$ and $\mathrm{GG_{ca}} \not\subset \mathrm{GG_{fa}}$. Finally, it holds that $\mathrm{GG_{ca}} \subset \mathrm{DG_c}$ and $\mathrm{GG_{ca}} \subset \mathrm{DG_f}$, because $\mathrm{GG_{ca}} \subset \mathrm{GG_{c1}}$ and $\mathrm{GG_{c1}} \subset \mathrm{DG_c} \subset \mathrm{DG_f}$.

Next we focus on $\mathrm{GG_{fa}}$. It is obvious that $\mathrm{GG_{fa}} \subset \mathrm{GG_{f1}}$ and that, in some cases, $\mathrm{GG_{fa}} \subsetneq \mathrm{GG_{f1}}$. The graph in Figure 2b shows that $\mathrm{GG_{c1}} \not\subset \mathrm{GG_{fa}}$, while the one in Figure 2c proves that $\mathrm{GG_{fa}} \not\subset \mathrm{DG_c}$. Since $\mathrm{GG_{fa}} \subset \mathrm{GG_{f1}}$ and $\mathrm{GG_{f1}} \subset \mathrm{DG_f}$, we conclude that $\mathrm{GG_{fa}} \subset \mathrm{DG_f}$.

Now let G be the graph in Figure 3a. This graph does not satisfy NIT and illustrates the changes with respect to the case in which this assumption is satisfied. That is, the graphs $\mathrm{NNG}(U;G)$, $\mathrm{UMST}(U;G)$, and $\mathrm{RNG}(U;G)$ contain edges (a,b), (b,c) and (c,a), while the graphs $\mathrm{GG_{ca}}(U;G)$, $\mathrm{GG_{c1}}(U;G)$, $\mathrm{GG_{fa}}(U;G)$, $\mathrm{GG_{f1}}(U;G)$, $\mathrm{DG_c}(U;G)$, and $\mathrm{DG_f}(U;G)$ only contain (a,b). Hence, none of the graphs in the first group is a subgraph of any graph in the second group.

Finally, analogous proofs to those in Section 3 show that the remaining relations of containment hold even if degeneracies are permitted. ⊠

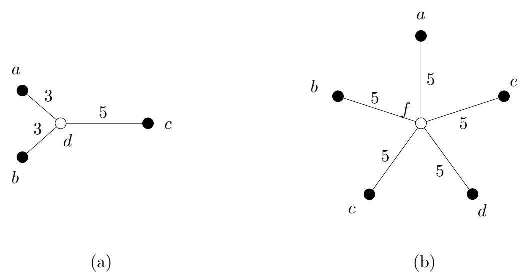

Figure 3: Two graphs not satisfying NIT. (a) The edge (b,c) belongs to $\mathrm{NNG}(U;G)$, $\mathrm{UMST}(U;G)$, and $\mathrm{RNG}(U;G)$, but does not belong to $\mathrm{GG_{ca}}(U;G)$, $\mathrm{GG_{c1}}(U;G)$, $\mathrm{GG_{fa}}(U;G)$, $\mathrm{GG_{f1}}(U;G)$, $\mathrm{DG_c}(U;G)$, and $\mathrm{DG_f}(U;G)$. (b) $\mathrm{NNG}(U;G)$, $\mathrm{UMST}(U;G)$, and $\mathrm{RNG}(U;G)$ are the complete graph on U, whereas $\mathrm{GG_{ca}}(U;G)$, $\mathrm{GG_{c1}}(U;G)$, $\mathrm{GG_{fa}}(U;G)$, $\mathrm{GG_{f1}}(U;G)$, $\mathrm{DG_c}(U;G)$, and $\mathrm{DG_f}(U;G)$ are the empty graph.

We now focus on the most important properties presented in Section 4.

The proof that $\mathrm{DG_f}(U;G)$ is the dual graph of $\mathrm{VD}(U;G)$ does not require any non-degeneracy assumption, so this result holds in all cases.

Let us now consider the size of the proximity graphs. If NIT is not satisfied, some of the proximity graphs might have more edges than the original graph. More precisely,

Theorem 6.2. *Let $G = (V,U,E)$ be a graph. The number of edges of $\mathrm{GG_{ca}}(U;G)$, $\mathrm{GG_c}(U;G)$, $\mathrm{GG_{fa}}(U;G)$, $\mathrm{GG_f}(U;G)$, $\mathrm{DG_c}(U;G)$, and $\mathrm{DG_f}(U;G)$ is at most e. The number of edges of $\mathrm{NNG}(U;G)$, $\mathrm{UMST}(U;G)$, and $\mathrm{RNG}(U;G)$ may be greater than e.*

Proof. Let (u_i,u_j) be an edge of $\mathrm{DG_f}(U;G)$. Then there is a point p of G such that $d_G(p,u_i) = d_G(p,u_j) < d_G(p,u_k)$ for all vertices $u_k \in U$ different from u_i and u_j.

First suppose that the point p is unique. If p is not an element of V, we assign the edge $(u_i, u_j) \in \mathrm{DG_f}(U; G)$ to the edge in G containing p. Otherwise let $u_i v_1 v_2 \ldots v_l p$ be a shortest path from u_i to p in G. Then we assign (u_i, u_j) to the edge $(v_l, p) \in G$.

Now suppose that there are several points of G satisfying the same condition as p. Let q be the one at smallest distance from u_i and u_j. Observe that $q \in V$. As in the previous case, if $u_i w_1 w_2 \ldots w_h q$ is a shortest path from u_i to q in G, we assign (u_i, u_j) to the edge $(w_h, q) \in G$.

We have just constructed an injective function that maps each edge in $\mathrm{DG_f}(U; G)$ to an edge in G. Hence the number of edges of $\mathrm{DG_f}(U; G)$ is at most e. Since $\mathrm{GG_{ca}}(U; G)$, $\mathrm{GG_{c1}}(U; G)$, $\mathrm{GG_{fa}}(U; G)$, $\mathrm{GG_{f1}}(U; G)$, and $\mathrm{DG_c}(U; G)$ are subgraphs of $\mathrm{DG_f}(U; G)$, their size is also bounded by e.

With regard to the remaining proximity graphs, consider the graph on Figure 3b, which does not fulfill NIT. Observe that $\mathrm{NNG}(U; G)$, $\mathrm{UMST}(U; G)$, and $\mathrm{RNG}(U; G)$ are the complete graph on the vertices of U, so they have ten edges, while the original graph has five. ⊠

Finally, the next two theorems include the proximity graphs that inherit the property of being planar or acyclic in the degenerate case. The proof of the first part of each statement can be obtained using the same approach as in Theorems 4.5 and 4.7 in Section 4. The graph in Figure 3b proves the second part of each statement.

Theorem 6.3. *Let $G = (V, U, E)$ be a planar graph. Then the graphs* $\mathrm{GG_{ca}}(U; G)$, $\mathrm{GG_{c1}}(U; G)$, $\mathrm{GG_{fa}}(U; G)$, $\mathrm{GG_{f1}}(U; G)$, $\mathrm{DG_c}(U; G)$, *and* $\mathrm{DG_f}(U; G)$ *are planar, whereas* $\mathrm{NNG}(U; G)$, $\mathrm{UMST}(U; G)$, *and* $\mathrm{RNG}(U; G)$ *may not be.*

Theorem 6.4. *Let $G = (V, U, E)$ be a tree. Then the graphs* $\mathrm{GG_{ca}}(U; G)$, $\mathrm{GG_{c1}}(U; G)$, $\mathrm{GG_{fa}}(U; G)$, $\mathrm{GG_{f1}}(U; G)$, $\mathrm{DG_c}(U; G)$, *and* $\mathrm{DG_f}(U; G)$ *are acyclic, whereas* $\mathrm{NNG}(U; G)$, $\mathrm{UMST}(U; G)$, *and* $\mathrm{RNG}(U; G)$ *may not be.*

The algorithms in the preceding section can be adapted to run under the presence of degeneracies yet we omit their descriptions which require, as usual, many details to be carefully handled.

References

[1] M. Abellanas and F. Harary, Delaunay graphs on a prescribed graph, Proc 15th European Workshop on Computational Geometry, Antibes, France, 1999, pp. 101–103.

[2] M. Abellanas, F. Hurtado, C. Icking, E. Langetepe, R. Klein, L. Ma, B. Palop, and V. Sacristán, Voronoi diagram for services neighboring a highway, Inform Process Lett 86(5) (2003), 283–288.

[3] J. Abello, P.M. Pardalos, and M.G. Resende, Handbook of massive data sets, Kluwer Academic Publishers, Dordrecht, 2002.

[4] H.-K. Ahn, H. Alt, T. Asano, S. Bae, P. Brass, O. Cheong, C. Knauer, H.-S. Na, C.-S. Shin, and A. Wolff, Constructing optimal highways, Internat J Found Comput Sci 20(1) (2009), 3–23.

[5] T.M. Chan, More algorithms for all-pairs shortest paths in weighted graphs, SIAM J Comput 39(5) (2010), 2075–2089.

[6] B. Chazelle, A minimum spanning tree algorithm with inverse-Ackermann type complexity, J ACM 47(6) (2000), 1028–1047.

[7] M. Erwig, The graph Voronoi diagram with applications, Networks 36(3) (2000), 156–163.

[8] M.L. Fredman and R.E. Tarjan, Fibonacci heaps and their uses in improved network optimization algorithms, J ACM 34(3) (1987), 596–615.

[9] T. Furuta, A. Suzuki, and K. Inakawa, The kth nearest network Voronoi diagram and its application to districting problem of ambulance systems, Discussion Paper, No.0501, Center for Management Studies, Nanzan University, 2005.

[10] S. Govindarajan, On locally Gabriel graphs, Submitted.

[11] F. Harary, M.S. Jacobson, M.J. Lipman, and F.R. McMorris, Abstract sphere-of-influence graphs, Math Comput Modelling 17(11) (1993), 77–83.

[12] F. Hurtado, R. Klein, E. Langetepe, and V. Sacristán, The weighted farthest color Voronoi diagram on trees and graphs, Comput Geom 27(1) (2004), 13–26.

[13] J.W. Jaromczyk and M. Kowaluk, A note on relative neighborhood graphs, Proc 3rd Annual Symposium on Computational Geometry, Waterloo, ON, 1987, pp. 233–241.

[14] S. Kapoor and X.-Y. Li, Proximity structures for geometric graphs, Internat J Comput Geom Appl 20(4) (2010), 415–429.

[15] G. Liotta, "Proximity drawings," Handbook of graph drawing and visualization, R. Tamassia (Editor), CRC Press, Boca Raton, to appear.

[16] K. Mehlhorn, A faster approximation algorithm for the Steiner problem in graphs, Inform Process Lett 27(3) (1988), 125–128.

[17] A. Okabe, B. Boots, K. Sugihara, and S.N. Chiu, Spatial tessellations: concepts and applications of Voronoi diagrams, John Wiley & Sons, Chichester, 2000.

[18] A. Okabe, T. Satoh, T. Furuta, A. Suzuki, and K. Okano, Generalized network Voronoi diagrams: concepts, computational methods, and applications, Int J Geogr Inf Sci 22(9) (2008), 965–994.

[19] Y. Ostrovsky-Berman, The transportation metric and related problems, Inform Process Lett 95(4) (2005), 461–465.

[20] S. Pettie and V. Ramachandran, An optimal minimum spanning tree algorithm, J ACM 49(1) (2002), 16–34.

[21] S. Pettie and V. Ramachandran, A shortest path algorithm for real-weighted undirected graphs, SIAM J Comput 34(6) (2005), 1398–1431.

[22] R. Pinchasi and S. Smorodinsky, On locally Delaunay geometric graphs, Proc 20th Annual Symposium on Computational Geometry, Brooklyn, New York, 2004, pp. 378–382.

[23] H. Samet, Foundations of multidimensional and metric data structures, Morgan Kaufmann, San Francisco, 2006.