Fine-Grained Location-Free Planarization in Wireless Sensor Networks

Dezun Dong, *Member*, *IEEE*, Xiangke Liao, Yunhao Liu, *Senior Member*, *IEEE*, Xiang-Yang Li, *Senior Member*, *IEEE*, and Zhengbin Pang

Abstract—Extracting planar graph from network topologies is of great importance for efficient protocol design in wireless ad hoc and sensor networks. Previous techniques of planar topology extraction are often based on ideal assumptions, such as UDG communication model and accurate node location measurements. To make these protocols work effectively in practice, we need extract a planar topology in a location-free and distributed manner with small stretch factors. The planar topologies constructed by current location-free methods often have large stretch factors. In this paper, we present a fine-grained and location-free network planarization method under ρ-quasi-UDG communication model with $\rho \geq 1/\sqrt{2}$. Compared with existing location-free planarization approaches, our method can extract a provably connected planar graph, called topological planar simplification (TPS), from the connectivity graph in a fine-grained manner using local connectivity information. We evaluate our design through extensive simulations and compare with the state-of-the-art approaches. The simulation results show that our method produces high-quality planar graphs with a small stretch factor in practical large-scale networks.

Index Terms—Wireless sensor networks, planarization, location-free, connectivity, fine grained, topological planar simplification

1 Introduction

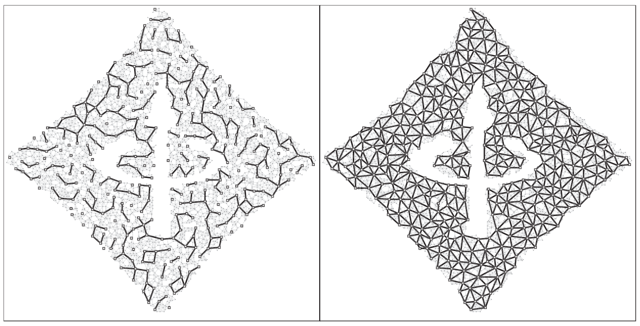
Nodes in wireless ad hoc and sensor networks are inherently placed in a geometric environment and can only communicate with nodes within a certain geometry neighborhood. The inherent geometry properties have been exploited to design a number of efficient protocols for wireless networks, such as geographic routing, topology discovery, etc. It is crucial to extract a planar topology from the communication graph while preserving the intrinsic network distances for the successful execution of many protocols. For example, in geometric routing (a.k.a geographic routing) protocols [1], [2], [3], [4], the faces of planar communication graph are used to perform perimeter routing, which guarantees packet delivery and greatly reduces the protocol complexity. In network localization schemes, planarized network topology helps to design efficient localization algorithms [5]. In topology discovery schemes [6], special planar substructures of network communication graph, i.e., boundary cycles, are extracted to locate communication holes, which contributes to the detection of faulty nodes and improves the load balancing and resilience of routing.

The most prominent approaches for addressing network planarization problem utilize the geometry locations of nodes [7], [8], [9], [10], [11]. In particular, the majority of those location-based algorithms [7], [8] are designed under communication models of unit disk graph (UDG), and a few ones [9], [10] make an effort to construct planar graphs in quasi unit disk graph (quasi-UDG) and extended graphs. The dependence on node locations, however, limits the applicability of those methods because acquiring location information is often practically difficult and expensive for large-scale networks. It is usually costly to equip every node with GPS devices to get accurate location measurement. For range-based and range-free localization methods, the problem is computationally NP-hard [12]. Those localization algorithms usually output probabilistic results as they suffer from error accumulation, flip ambiguity, etc. It is, thus, important to relax the assumption on location measurements to enhance the applicability of algorithms in resource-limited wireless networks.

Recently, location-free planarization has received considerable attention. Funke and Milosavljevic [13] propose a distributed method to find a provable planar graph, called combinatorial Delaunay map (CDM). Zhang et al. [14] formalize network planarization as the NP-hard bipartite planarization problem, and propose a layer-by-layer planarization method. Spanning ratio (or distortion) is widely recognized as an important metric to measure the quality of a planar subgraph [8]. Though current location-free solutions [13], [14] shed some light on the challenging issue of location-free planarization, given the large distortion of those planar structures constructed by them, they cannot be easily applied in those applications that demand

- *D. Dong and X. Liao are with the School of Computer, National University of Defense Technology, Changsha, Hunan 410073, P.R. China, and also with the National Laboratory for Paralleling and Distributed Processing, National University of Defence Technology, Hunan 410073, P.R. China. E-mail: {dong, xkliao}@nudt.edu.cn.*
- *Y. Liu is with the School of Software, TNLIST, Tsinghua University, Bejing 100084, China, and also with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong. E-mail: liu@cse.ust.hk.*
- *X.-Y. Li is with the Department of Computer Science, Illinois Institute of Technology, 10 West 31st Street, Chicago, IL 60616. E-mail: xli@cs.iit.edu.*
- *Z. Pang is with the School of Computer, National University of Defense Technology (NUDT), Changsha, Hunan 410073, P.R. China. E-mail: zbpang@163.com.*

Manuscript received 13 Jan. 2011; revised 9 Sept. 2011; accepted 14 Feb. 2012; published online 28 Feb. 2012.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2011-01-0016. Digital Object Identifier no. 10.1109/TMC.2012.57.



(a) A CDM by previous work [13] (b) A TPS by this design

Fig. 1. Demo CDM and TPS in a network deployed in a complex geometric region. As a dense sampling of the network, 367 nodes (approximate one of ten total network nodes) are selected as the vertices of CDM and TPS. Bold lines denote edges of CDM and TPS. CDM is disconnected into 96 components while TPS is a well-connected graph with a small stretch factor.

a high-quality planar topology. For example, Zhang et al.'s method [14] requires that the network be deployed in regions with square-like shapes. When the network deployment region has feature-rich shapes or holes, planar graphs extracted by their method could have arbitrarily large distortion. Comparatively, Funke and Milosavljevic's method does not put any conditions on the shape of network deployment regions. However, it outputs a well-connected CDM only when each Voronoi tile of CDM has a large diameter: their theoretical result requires the diameter be at least 290 (in hop-number metric), although their simulation results show that their method still works when the diameter of each Voronoi tile is at most 10. Under such circumstances vertices of CDM are indeed a sparse sampling of the network, which in turn results in a large stretch factor of the constructed topology. If we force vertices of CDM to be a dense sampling of the network, e.g., the diameter of each Voronoi tile is at most 4, CDM will be disconnected and contain many connected components, as illustrated in the example shown in Fig. 1a.

In this design, we make the first attempt toward location-free planarization such that the constructed planar structure has a small stretch factor for most inputs. We construct a high-quality virtual planar backbone of the network, called *topological planar simplification* (TPS). Our method first performs a uniform sampling in the network and constructs a simplified structure, called restricted witness graph (RWG), that is a spanner, but maybe not a planar graph. We construct a minimal connectivity subgraph of the network graph to represent RWG, called underlying representation. We exploit underlying representations of RWG to acquire the intrinsic geometric structure of RWG from the underlying deployment domain. Based on underlying representations, we define conflicts between RWG edges to capture all possible edge potentially causing nonplanarity of RWG. Our conflict resolution techniques locally search proper underlying paths and dynamically eliminate a separable conflict. Although both Funke and Milosavljevic's and our methods extract planar graphs from similar background graphs, the high-level ideas for planar graph constructions are different. In their method, the planar drawing of CDM is generated from a set of deterministic safe paths. However, our work uses underlying representations to construct a planar drawing of TPS in a dynamic fashion such that edges not in CDM can often be transformed into conflict-free edges and be selected into TPS. As a result, our TPS is at least as good as a supergraph of CDM, and outperforms CDM on many practical instances. In this example shown in Fig. 1, the TPS extracted from the same network is greatly superior to the CDM.

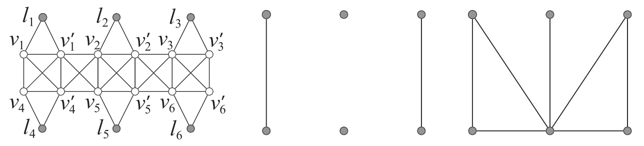
The main contributions of this work are as follows: we propose a practical distributed algorithm that does not use any location, angular, or distance information but merely connectivity information to extract a provable planar topology, i.e., TPS, under ρ-quasi-UDG communication model with $\rho \geq 1/\sqrt{2}$. We prove that TPS is a connected planar substructure for all possible inputs under quasi-UDG models. Although we currently cannot guarantee the TPS constructed by our algorithm has constant stretch factors for arbitrary inputs, we provide a simple condition to test the output of our algorithm for correctness. If this condition is satisfied, a constant planar spanner is definitely generated by our algorithm. We evaluate the performance of our method through extensive simulations. The simulation results show that the TPS built by our algorithm always has a small stretch factor in practical networks where nodes are deployed in a uniformly random distribution with different configurations, including varying the shapes of deployment regions, node density, network scale, etc. Our method is, thus, successful in producing high-quality planar graphs in practice. Hence, our design achieves a *fine-grained* location-free planarization. TPS is expected to improve the performance of many applications built upon CDM graphs greatly, e.g., geometric routing protocols [4]. It can also be beneficial to various applications, such as network localization, segmentation, and topology discovery [6], [15].

The remainder of this paper is organized as follows: we discuss related work in Section 2, and introduce the problem formulation in Section 3. Section 4 presents the algorithm of topological planar simplification. Section 5 further discusses details of this design. We analyze the correctness and performance of our algorithm in Section 6 and present the evaluation in Section 7, followed by the conclusion in Section 8.

2 Related Work

We can divide existing work on network planarization into two categories: *location-based* and *location-free*. For location-based planarization, most efforts focus on finding planar structures for geometric UDGs, which is widely used in topology control for wireless ad hoc networks. Some well known structures include Gabriel graph (GG), relative neighborhood graph (RNG), local minimum spanning trees (LMSTs), restricted Delaunay graph (RDG) [7], localized Delaunay graph (LDel) [8], etc. There also exist some location-based methods seeking planar graphs in quasi-UDGs or extended graphs, such as GridGraph [10] and CLDP [9]. Please refer to good surveys by Wang [11] for more location-based methods. We here pay more attention to location-free methods [13], [14] as our work is in this category.

We use a simple example in Fig. 2 to explain the main idea of Funke and Milosavljevic's method [13]. Their



(a) Connectivity graph (b) CDM (c) TPS

Fig. 2. A simple example for CDM and TPS.

method has two steps. In the first step, some nodes are selected as landmarks from connectivity graph, labeled by l_i in Fig. 2a, i from 1 to 6. Each nonlandmark node is affiliated with a landmark closest to itself in terms of hop count. The connectivity graph is partitioned into a collection of disjoint subsets, Voronoi tiles. Further a combinatorial Delaunay graph (CDG) is built to capture the adjacency between the tiles. In the example, each tile includes three nodes l_i, v_i and v_i', i from 1 to 6. Those techniques in their first step, including network sampling, partition and CDG construction, have been proposed and used in previous work [4], [16], [17]. A similar first step also is adapted in our method. The key contribution of Funke and Milosavljevic's work is the techniques that extract a planar CDM graph from CDG. In particular, each landmark l_i is a vertex of CDM. An edge (l_i, l_j) is added to CDM if the following two conditions are satisfied: 1) there exists a path from l_i to l_j in the network that consists of a sequence of nodes associated with l_i followed by a sequence of nodes associated with l_j; 2) one-hop neighbors of the path only contain nodes associated with landmark l_i or l_j. CDM is guaranteed to be planar in ρ-quasi-UDG with $\rho \geq 1/\sqrt{2}$. Although CDM is a good structure to approximate global network skeletons, CDM is often disconnected to cause large distortions when tile sizes are relatively small. In this example shown in Fig. 2b, CDM contains four connected branches. Comparatively, our method explores effective new techniques to planarize the network in a fine-grained manner.

Zhang et al. [14] planarize a square-shaped network under realistic models with nonuniform transmission ranges. The main idea of their method is to label network nodes in layers, then planarize the network in a layer-by-layer manner through by formulating it as NP-hard bipartite planarization problem. There are two main shortages in their method. First, the method mainly works in networks that have regular square-shaped deployment region. It is hard to be extended into networks with arbitrary shapes, such as irregular outer boundary or inner holes. Two shortest path trees built by their method could only cover a small portion of the network with complex shapes. Consequently, the found planar graph will inevitably have a large distortion, as explained in Section 7.1. Second, they present centralized fixed parameter tractable (FPT) algorithms for the NP-hard bipartite planarization problem, which essentially requires the global connectivity information and incurs high complexity of communication and computation. It remains unknown how to perform their algorithms in an efficient distributed manner.

3 Problem Formulation

We present network assumptions and formulate the problem of topological planar simplification. We consider a collection of nodes deployed over a plane region. Each node has a unique identity (ID). Nodes are only capable of communicating with other nodes in its proximity. We assume that the coordinates of nodes are unavailable, in the sense that nodes can determine neither distances nor orientations. This makes our approach robust to situations where geometry information is missing or only partially available. We extract planar network topology in a connectivity graph G, where vertices and edges identify the nodes and communication links, respectively. Connectivity graph is far from a general graph in spite of missing location information, and has its inherent geometry properties. The quasi-UDG, known as generalized UDG model, can reflect the proximity and radio irregularity, and better capture the characteristics of wireless networks than UDG, so that it is widely used to model wireless ad hoc and sensor networks [10], [13], [18].

3.1 Graph Notation and Terminology

We introduce some graph notations used in this work. Let H be a simple graph with vertex set $V(H)$ and edge set $E(H)$. Given a vertex or edge set X in H, $H[X]$ denotes the subgraph of H induced by X. Given two vertex sets X and Y in H, we use $d_H(X, Y)$ to denote the minimum hop-number distance in H between any one vertex in X and any vertex in Y. In the rest we will also write $d_H(X, Y)$ as $d_H(H[X], Y)$, $d_H(X, H[Y])$ or $d_H(H[X], H[Y])$ for the convenience. The diameter $D(H)$ of H is the maximum distance between any two vertices in H. This work focuses on graph spanner instead of geometrical spanner. A general definition for graph spanner is given as follows: An (α, β)-*spanner* [19] of H is a subgraph H' such that $d_{H'}(u, v) \leq \alpha d_H(u, v) + \beta$, for any two vertices u, v in H. If $\alpha = 1$, the spanner is called an *additive* β-*spanner*. If $\beta = 0$, this definition reverts to the usual definition of a *multiplicative* α-*spanner*, and α is the *stretch factor*. A node subset of a graph is a *k-hop maximal independent set* (kMIS) if it meets the following two conditions: 1) the pairwise distances of the nodes in the subset are all greater than k; 2) adding any extra nodes into the set will break the first condition.

Graph H is a ρ-quasi-UDG with parameter $0 < \rho \leq 1$ if there exists an embedding $\varepsilon : V \rightarrow \mathbb{R}^2$, which maps the vertices of H into the euclidean plane, such that for any two vertices u and v in $V(H)$, 1) if the euclidean distance $|\varepsilon(u)\varepsilon(v)| \leq \rho$, then (u, v) is an edge in $E(H)$; 2) if $|\varepsilon(u)\varepsilon(v)| > 1$, then (u, v) is not an edge in $E(H)$; 3) and if $\rho < |\varepsilon(u)\varepsilon(v)| \leq 1$, (u, v) can be or not an edge in $E(H)$. An embedding ε is called a *realization* of H. When $\rho = 1$, a quasi-UDG is a UDG. This study uses *combinatorial* quasi-UDGs, where only a collection of vertices and connectivity among vertices are known, different with *geometric* quasi-UDGs, where a realization is also given.

Let $p = H[V_p]$ be a subgraph of H induced by a sequence of distinct vertices $V_p = \{v_1, v_2, \ldots, v_m\} \subseteq V(H)$ such that $E(p) = \{(v_i, v_{i+1}) : i \in [1, m-1]\}$ and no other edges exist between any two of these vertices. We refer to p as a *chordless path* of length $|p| = m$. A *shortest path tree* is a subgraph of a given graph constructed such that the total distance from a selected root node to other nodes is minimized. A simple cycle C is a connected subgraph of H and the degree of each vertex in C is two. A cycle C can be identified by its incidence vector $b(C) = (b_1, b_2, \ldots, b_i, \ldots)$, for $i \in [1, |E(H)|]$, with $b_i = 1$ iff $e_i \in E(C)$ and $b_i = 0$ iff $e_i \notin E(C)$. The length $|C|$ of cycle

C is the number of its edges, $|E(C)|$. The vector space over $GF(2)$ generated by the incidence vectors of cycles is called the *cycle space* of H, where $GF(2)$ denotes the Galois field over $\{0,1\}$. The addition of two cycles C_1 and C_2 is the modulo 2 addition of their incidence vectors. It corresponds to the symmetric difference $C_1 \oplus C_2 = (E(C_1) \bigcup E(C_2)) \setminus (E(C_1) \bigcap E(C_2))$. Given a cycle set $\mathcal{C} = \{C_i : i \in [1,n]\}$, the cycle sum of $\mathcal{C}$ is $\sum \mathcal{C} = C_1 \oplus C_2 \oplus \cdots \oplus C_n$. A *cycle basis* $\mathcal{B}$ of H is a basis of $\mathcal{C}_H$. Given a cycle C and a cycle set $\mathcal{C}$ in graph H, if C is the sum of cycles in $\mathcal{C}$, $C = \sum \mathcal{C}$, $\mathcal{C}$ is a *cycle partition* of C in G [20]. We use $|\mathcal{C}|_{max}$ to denote the length of the longest cycle in $\mathcal{C}$. Given two paths p_1 and p_2 sharing the same endpoints in H, let $E' = (E(p_1) \bigcup E(p_2)) \setminus (E(p_1) \bigcap E(p_2))$ be the symmetric difference of edges in p_1 and p_2. The *concatenation* of p_1 and p_2, denoted by $\widehat{p_1p_2}$, is a subgraph of H induced by E'. Graph $\widehat{p_1p_2}$ includes either a simple cycle or a union of edge-disjoint simple cycles.

3.2 Topological Planar Simplification

Spanner property is widely recognized as an important metric to measure the quality of a planar subgraph [8]. The two properties of planarity and spanner, however, are often in conflict with each other. For example, given any $0 < \rho < 1$, there exists ρ-quasi-UDGs such that it is theoretically infeasible to construct a planar multiplicative spanner with constant stretch factor in these ρ-quasi-UDGs. Though planar (α,β)-spanner in combinatorial quasi-UDGs still has not been sufficiently studied, it is known that a ρ-quasi-UDG, $\rho \geq 1/\sqrt{2}$, has a planar (α,β)-spanner with bounded constant α and β. This is due to the fact that a $1/\sqrt{2}$-quasi-UDG has an important *"link-crossing"* property. That is, given a $1/\sqrt{2}$-quasi-UDG H, if two edges (u,v) and (x,y) in H cross in a valid realization of H, there exist at least three edges between nodes u,v,x,y in H. By default, the spanner mentioned in the rest refers to its general definition. This study focuses on extracting a planar structure from the connectivity graph G such that a planar (α,β)-spanner can be efficiently constructed from this structure. We formalize this problem as topological planar simplification as follows.

Definition 1 (Topological planar simplification). *Given a connectivity graph $G = (V,E)$, a* topological planar simplification *of G is a planar graph $G' = (V',E')$, where V' is a subset of V, and each edge $(u,v) \in E'$ corresponds to a path p connecting u and v in G. G' is called a (α,β)-*TPS *of G, if there exist two constants α and β such that $d_{G'}(u,v) < \alpha \cdot d_G(u,v)$ for $u,v \in V'$, and $d_G(v,V') \leq \beta$ for any $v \in V$.*

4 Topological Planar Simplification Algorithm

We present a distributed algorithm to construct a TPS using connectivity information. Our algorithm mainly includes three components: 1) *construct RWG with underlying representation*, 2) *calculate maximal conflict-free edge set*, and 3) *perform conflict resolution*. We next describe the details of each component. We use an example illustrated in Fig. 3 to explain this design. Given an arbitrary connectivity graph G, such as the one shown in Fig. 3a, our algorithm aims at extracting a TPS graph G_{TPS} from G. Square nodes and bold-line edges in Fig. 3i show one TPS found by our method.

4.1 Construct RWG with Underlying Representation

We first introduce the definition of restricted witness graph. RWG is inspired by the *witness complex* [21]. Witness complex becomes popular in wireless network since a special case of witness complexes, i.e., combinatorial Delaunay triangulation (CDT), is used as a tool for landmark-based routing [13], [16]. Witness complex, however, can generally have an arbitrary stretch factor since the size of its tile is not bounded. This motivates us to define RWG to bound the size of tiles. RWG is a good structure for building a constant spanner, as discussed in Section 6.

Definition 2 (Restricted witness graph). *Given a graph G, and two positive constants δ and λ, a (δ,λ)-*restricted witness graph *G_R is defined on G as follows: 1) each vertex v_i of G_R corresponds to a vertex subset V_i of $V(G)$ such that $D(G[V_i]) \leq \delta$ and $\bigcup_{i=1}^{|V(G_R)|} V_i = V(G)$; 2) an edge (v_i,v_j) of G_R exists if and only if $d_G(V_i,V_j) \leq \lambda$.*

We next present a practical construction for a specific RWG used in this work. This procedure mostly resembles the construction of CDG used in previous work [13]. Given connectivity graph G, we select a kMIS in G, denoted by $V_{k\text{MIS}}$. Throughout this paper we fix k to be a small constant 2 to build a dense sampling of network. We use $V_{k\text{MIS}}$ as landmarks to partition G into proper tiles. Specifically, for each vertex v in G but not in $V_{k\text{MIS}}$, v is affiliated with the landmark l, $l \in V_{k\text{MIS}}$, such that for any other landmark l' in $V_{k\text{MIS}}$, $d_G(v,l) \leq d_G(v,l')$ and the node ID of l is less than that of l' if $d_G(v,l) = d_G(v,l')$. We, thus, partition the graph nodes into disjoint subsets (or tiles). No ties are permitted during this RWG construction, since in practice that makes the resulting RWG more planar. The edges of RWG are further obtained according to whether nodes between two tiles share at least one common edge. It is not difficult to see that all these operations can be implemented in a distributed manner. The above construction achieves a $(2k,1)$-RWG. We use G_R to denote the built RWG. A $(2k+1,2k)$-spanner of G can be built from G_R, as shown in Lemma 5. Fig. 3b illustrates the above operations. Nodes in the same tiles are connected by light lines and labeled with the same marks. Dark lines show the edges that witness the adjacency between tiles. In Fig. 3c, square landmarks are connected by bold lines to indicate the RWG.

We next construct a minimal subgraph of G to represent the RWG G_R, called an *underlying representation*, presented in Definition 3. The construction of an underlying representation for RWG is an important step to identify edges that lead to nonplanarity of RWG. This point will be explained in Section 4.2.

Definition 3 (Underlying representation). *An* underlying representation *(L,P) for RWG G_R is a collection of landmark nodes L and paths P in G. L denotes* underlying nodes *in one-to-one correspondence with the vertices of G_R, and P denotes* underlying paths *in one-to-one correspondence with the edges of G_R. For each edge $e = (l,l')$ in G_R, its underlying path $p_e \in P$ is defined as one chordless path in G*

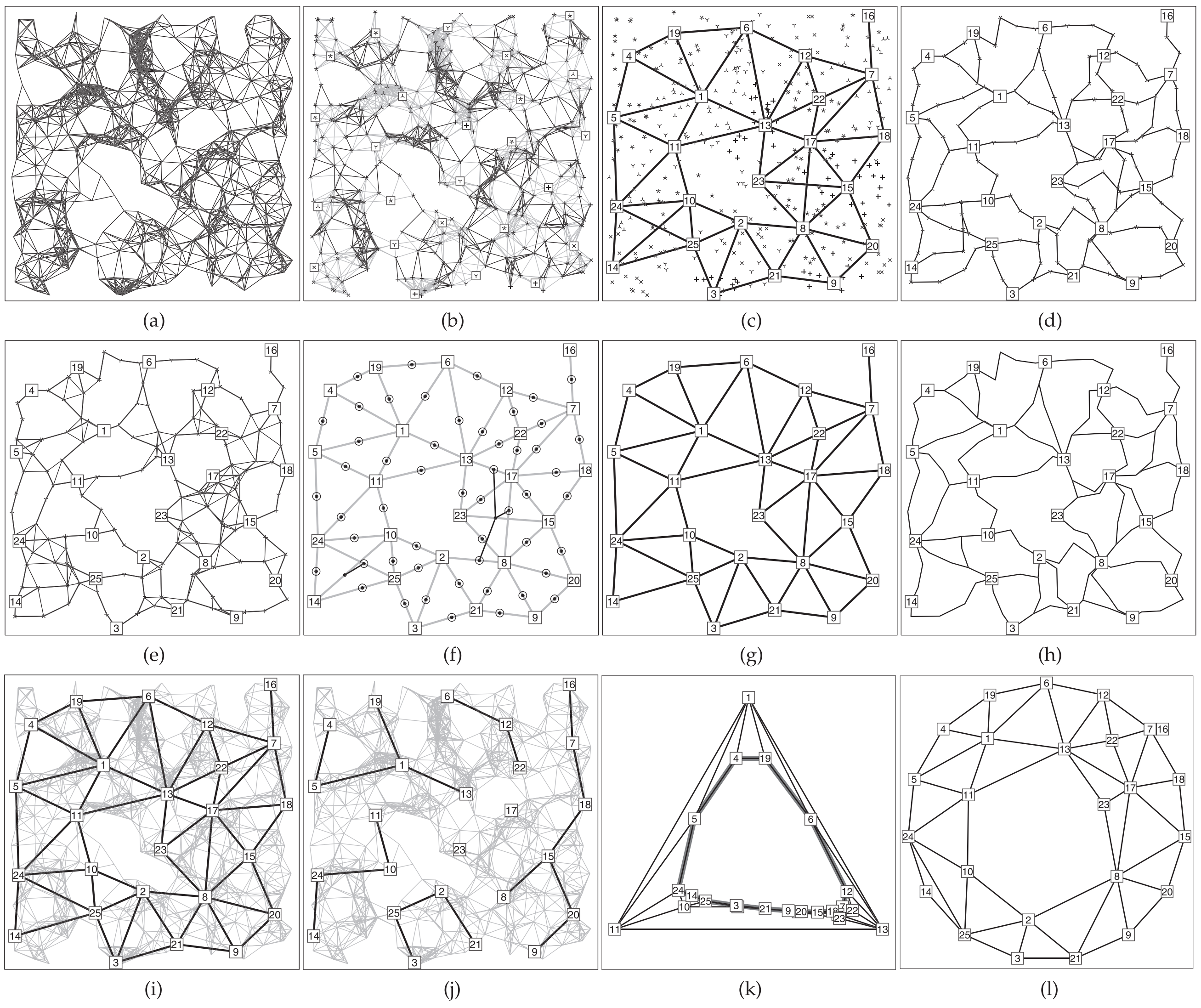


Fig. 3. Topological planar simplification and embedding. (a) The connectivity graph G with 400 nodes and average node degree 10.69, following 0.75-quasi-UDG model. (b) RWG construction. Nodes in the same tile are labeled with the same flag. Light lines are edges among nodes in the same tile, and dark lines show edges between nodes in different tiles. (c) The RWG, whose vertices and edges are denoted by squares and lines respectively. (d) Underlying paths, denoted by lines. (e) The subgraph of G induced by all nodes in the underlying paths. (f) Conflict graph, whose vertices and edges are dots and dark lines, respectively. The circles show a maximal independent set. (g) Maximal conflict-free graph of RWG. (h) Underlying paths of maximal conflict-free graph. (i) The obtained TPS after conflict resolution for remained edges. (j) The CDM extracted in the same network by previous work [13]. (k) and (l) are two planar embeddings of TPS.

which connects landmarks l and l' and only consists of nodes associated with landmarks l and l'.

An underlying representation for RWG G_R obviously exists, and can be easily found by using local connectivity. A landmark node only needs to gather the connectivity within k hops and interacts with landmarks of neighboring tiles to construct underlying paths locally. In practice, one edge of G_R often has many candidate underlying paths. To reduce the cost of recording an underlying path, we select one of the shortest underlying paths. Fig. 3d shows one underlying representation for the RWG shown in Fig. 3c.

As a remark, RWG in Definition 2 is defined in a very general manner. RWG does not require that each node in a tile must be closest to the landmark in its tile as CDG, which potentially makes it more flexible to construct RWG than CDG, as discussed in Section 5.1.

4.2 Calculate Maximal Conflict-Free Edge Set

The RWG is nonplanar in most practical instances, e.g., the one shown in Figs. 3c. In this section, we explore the underlying representation of RWG to extract a maximal conflict-free edge set from RWG G_R. The subgraph of G_R induced by conflict-free edges is a planar graph, because we can use the underlying paths of conflict-free edges to construct a planar drawing, as explained in Lemma 4. We next introduce contiguous paths and conflicts among RWG edges in Definitions 4 and 5, respectively.

Definition 4 (Contiguous paths). *Given two paths p_1 and p_2 in G, p_1 and p_2 are* contiguous *if the distance between p_1 and p_2 in G is not greater than 1, $d_G(p_1, p_2) \leq 1$.*

In other words, p_1 and p_2 are contiguous if p_1 contains at least one node v such that $v \in V(p_2)$ or v is neighboring to at least one node in p_2.

Definition 5 (Maximal conflict-free edge set). *Given an underlying representation (L, P) of RWG G_R in G, two edges $e = (l_e, l'_e)$ and $f = (l_f, l'_f)$ in G_R are in* conflict *with respect to the underlying representation (L, P) if their endpoints do not share any underlying node in L, $\{l_e, l'_e\} \bigcap \{l_f, l'_f\} = \emptyset$, and their underlying paths p_e and p_f in P are contiguous. Otherwise, e and f are* conflict-free *with respect to (L, P). An edge set E' in G_R is* conflict-free *if any two edges in E' are conflict-free with respect to (L, P), and E' is* maximal *if it is not a proper subset of any other conflict-free edge set in G_R.*

We construct a maximal conflict-free edge set in RWG G_R. For each underlying path, we first determine other underlying paths that are contiguous to it. We thus obtain the conflict relationship between any two edges in G_R. Based on these conflict relationships, we further construct a conflict graph G_X, as follows: One vertex in G_X corresponds to one edge in G_R, and one edge in G_X represents the conflict relationship of two edges in G_R. Finally, we build a maximal independent set V_{MIS} in G_X, and accordingly we obtain an edge set $E_{V_{\text{MIS}}}$ in G_R. Edges $E_{V_{\text{MIS}}}$ are in one-to-one correspondence with vertices V_{MIS}. $E_{V_{\text{MIS}}}$ is a maximal conflict-free edge set in G_R. All these operations in this component, including determining the conflict relationship between two RWG edges, computing a maximal independent set, and, etc., can be easily implemented in a distributed manner. We finally add all vertices in G_R and edges $E_{V_{\text{MIS}}}$ to G_{TPS}, that is, $V(G_{TPS}) = V(G_R)$, $E(G_{TPS}) = E_{V_{\text{MIS}}}$.

Fig. 3e shows the subgraph of G induced by all nodes in the underlying paths. Conflict graph G_X is shown in Fig. 3f, where one dot identifies an edge in G_R, one dark line represents a conflict relationship, and the maximal independent set V_{MIS} is denoted by circle nodes. Figs. 3g is the maximal conflict-free edge set $E_{V_{\text{MIS}}}$ in G_R.

4.3 Perform Conflict Resolution

In this component, we deal with the remaining edges that cannot be added into the maximal conflict-free edge set, denoted by $E_L = E(G_R) \backslash E(G_{TPS})$. Each edge in E_L conflicts with at least one edge in $E(G_{TPS})$ in the current underlying representation. Some conflicts are caused by real crossing underlying paths, and others may be created due to the improper selection of underlying representation. For example, the conflict between edges $(17, 8)$ and $(23, 15)$ corresponds to real edge intersection in Fig. 3f. Nevertheless, the conflict between edges $(13, 17)$ and $(23, 15)$ is noncritical and can be resolved by using proper underlying representations. We next present techniques that resolve those conflicts caused by unfavorable underlying representations or real intersections. In particular, we separate the non-critical conflicts by calculating new underlying paths, or try to relax real conflicts such that stretch factor is as small as possible while planarity is preserved.

We start with a simple scenario to introduce the problem of conflict resolution. Let e be one edge in E_L that conflicts with only one edge f in G_{TPS} in the current underlying representation (L, P). Suppose we can find another underlying path p'_e for e to replace its original underlying path p_e in P and upgrade the current underlying representation (L, P) to (L, P'). If e does not conflict with f in the new underlying representation (L, P') and does not come into conflict with other edges, clearly a new conflict-free edge set is found, and edge e can be added to G_{TPS}. However, it is often infeasible to find such a good underlying path for e. Most new underlying paths for e will cause new conflicts with other edges. Let f be one edge in conflict with e. The underlying path for f may also need to be modified to make e and f conflict-free. The modified underlying path for f can further cause new conflicts. The problem becomes more complicated when edge e conflicts with multiple edges in G_{TPS}, for it becomes more difficult to eliminate multiple conflicts simultaneously. In these circumstances we wonder whether edge e can still be added to G_{TPS} without destroying the planarity of G_{TPS}. We thus propose a simple method to resolve each conflict individually. In particular, for each edge e' in conflict with e in G_{TPS}, if we can find proper separable underlying paths for both e' and e to make them conflict-free, we can add e into G_{TPS} safely. We next present the definition of cycle-homotopy paths and separable conflicts.

Definition 6 (Cycle-homotopy paths). *Given two paths p_1 and p_2 with the same endpoints in G and a positive integer ℓ_0, p_1 and p_2 are of ℓ_0-*cycle homotopy *in G, denoted by $p_1 \simeq p_2$, if the concatenation $\widehat{p_1 p_2}$ of p_1 and p_2 admits a cycle partition $\mathcal{C}$ in G, $\widehat{p_1 p_2} = \sum \mathcal{C}$, such that each cycle in $\mathcal{C}$ has the length at most ℓ_0, $|\mathcal{C}|_{max} \leq \ell_0$.*

In this paper, we fix ℓ_0 be a small constant 4, i.e., $\ell_0 = 4$. For an edge e in RWG G_R, we use $U(e)$ to denote the set of all underlying paths of e in G.

Definition 7 (Separable conflict). *Given two edges e_1 and e_2 in G_R, let $p_1, p'_1 \in U(e_1)$ and $p_2, p'_2 \in U(e_2)$ be paths in G. p_1 and p_2 are* separable paths *and the conflict between edges e_1 and e_2 is* separable, *if $p_1 \simeq p'_1$, $p_2 \simeq p'_2$, and p_1 and p_2 are contiguous while p'_1 and p'_2 are not contiguous.*

Take the network in Fig. 3 for example. We test the separability of conflicts for the two edges left $(14, 10)$ and $(23, 15)$. They conflict with $(24, 25)$ and $(17, 8)$, respectively. The test result shows that both the conflict between $(14, 10)$ and $(24, 25)$ and the conflict between $(23, 15)$ and $(17, 8)$ are not separable. Thus, the two edges left cannot be added to the current TPS G_{TPS}. We here explain the effectiveness of separable-conflict testing as follows: in this example, we can verify that both the conflict between $(13, 17)$ and $(23, 15)$ and the one between $(13, 17)$ and $(23, 15)$ are separable. Hence, if edge $(23, 15)$ is put into G_{TPS} instead of edges $(13, 17)$, $(2, 8)$, and $(17, 8)$ in the step of calculating maximal conflict-free edge set, $(13, 17)$ and $(2, 8)$ will be added to G_{TPS} through separable-conflict testing in this step.

After separable conflict resolution, the edges left are mainly the ones causing nonplanarity of RWG. In this example, $(14, 10)$ and $(23, 15)$ are the remaining edges, and their underlying paths intersect with underlying paths of other edges in G_{TPS}, which means that adding any of them into G_{TPS} has the risk of destroying the planarity of G_{TPS}. In the rest, we refer to the edges potentially causing nonplanarity of RWG as *risky edges* in RWG for conciseness. We apply a simple principle of *lazy adding* for the risky edges. Given a remaining edge $e = (u, v) \in E(G_R) \backslash E(G_{TPS})$, our method locally tests whether there exists a path p in G_{TPS}, such that p connects u and v, and the length of p is bounded by a constant μ. If so, edge (u, v) can be ignored safely, because the stretch ratio of G_{TPS} can still be bounded by a constant without adding (u, v) to G_{TPS}. In this paper, we fix the length of an alternate path to be a small constant. Our extensive simulations verify that $\mu = 3$ is enough to eliminate all remaining edges in practical networks where nodes are deployed in a uniformly random distribution with different configurations, including varying the shapes of deployment regions, node density, etc. If unfortunately we cannot find an alternate path of bounded length for one remaining edge, more techniques can be used to handle the special cases, as discussed in Section 5.

4.4 Putting It All Together

We now have all the components to build the whole algorithm and analyze the time complexity of this design. Our algorithm first generates an RWG from the connectivity graph, and then modifies RWG to obtain a TPS while keeping small distortion. Algorithm 1 summarizes the main steps of our algorithms.

Algorithm 1. Topological Planar Simplification

```
Input: Connectivity graph $G$
Output: A topological planar simplification $G_{TPS}$ for $G$
 1: Construct restricted witness graph $G_R$. Construct a
    $k$MIS $V_{k\mathrm{MIS}}$ in $G$, $k = 2$. Use $V_{k\mathrm{MIS}}$ as landmarks to
    partition $V(G)$ into a collection of node-disjointed tiles.
    Edges of $G_R$ are obtained according to whether nodes
    among two tiles share at least one common edge in $G$.
 2: Construct an underlying representation $(L, P)$ for
    $G_R$. $L := V_{k\mathrm{MIS}}$, $P := \emptyset$.
 3: for each edge $e = (l, l')$ in $E(G_R)$ do
 4:     Build one shortest path $p_e$ in $G$ that connects
        landmarks $l$ and $l'$ and consists only of nodes
        associated with landmarks $l$ and $l'$. Add $p_e$ into $P$.
 5: end for
 6: Build a conflict graph $G_X$ for $G_R$ with respect to $(L, P)$.
    Construct a maximal independent set $V_{\mathrm{MIS}}$ in $G_X$, and
    obtain maximal conflict-free edge set $E_{V_{\mathrm{MIS}}}$
    in $G_R$. $V(G_{TPS}) := V(G_R)$, $E(G_{TPS}) := E_{V_{\mathrm{MIS}}}$.
 7: for each edge $e = (v_i, v_j)$ in $E(G_R) \backslash E(G_{TPS})$ do
 8:     Add $e$ to $E(G_{TPS})$, if the conflict between $e$ and any
        edge in $E(G_{TPS})$ is separable.
 9: end for
10: if $E_L = E(G_R) \backslash E(G_{TPS})$ is not empty then
11:     Perform non-separable conflict resolution for each
        edge in $E_L$, and update $G_{TPS}$.
12: end if
13: Output $G_{TPS}$.
```

In particular, in the first component, we select a kMIS in G, $k = 2$, as landmarks to partition G into proper tiles, and then construct an RWG G_R, as shown in Figs. 3b and 3c. Let Δ be the maximum node degree and n be the total number of the nodes in the network. The best distributed deterministic MIS algorithm [22] runs in $\mathcal{O}(\log \Delta \log^* n)$ time in growth-bounded graph (GBG) model. GBG formulates a general family of graphs covering UDGs and quasi-UDGs. To affiliate each node with a landmark, a simple option is to use Dijkstra's algorithm to calculate a single-source shortest path tree for each landmark simultaneously. Each landmark l can perform a restricted flooding to construct its shortest path tree, whose height is bounded by k and root is l. Dijkstra's algorithm performs at most $\mathcal{O}(\Delta^{k+2})$ number of operations. Hence, this component requires

$$\mathcal{O}(\max\{\log \Delta \log^* n, \Delta^{k+2}\})$$

time at most. RWG is a good structure to construct a constant spanner of network graph, but usually not a planar graph. We need to locate the set of edges breaching the planarity of RWG, and extract a planar graph while keeping the distortion as small as possible. Without location information, it becomes difficult to locally determine those edges that cause nonplanarity of RWG. This design exploits the fact that the communication graph is not an arbitrary graph but has its intrinsic geometric structure derived from the underlying deployment domain. We extract a subgraph from G as the underlying representation of RWG. Each edge of RWG corresponds to an underlying path in the original graph, as illustrated in Fig. 3d.

In the second component, we construct a high-quality planar subgraph of RWG by combining the techniques of topological graph theory and the geometrical properties of quasi-UDG. To capture all possible edges causing nonplanarity of RWG, we introduce the conflict among RWG edges. We build a conflict graph for RWG and thus construct a maximal conflict-free edge set in RWG, as shown in Figs. 3f and 3g. The subgraph induced by conflict-free edges is guaranteed to be a planar graph, because we can use the underlying representation of those conflict-free edges to construct a planar drawing, as illustrated in Fig. 3h. Time cost of the second component is dominated by the MIS computation in the conflict graph for RWG. From Lemma 6, we know that the maximum node degree of G_R is bounded by a constant. Hence, it is not difficult to show that the maximum node degree of a conflict graph is also bounded by a constant. So, an MIS in the conflict graph can be computed in $\mathcal{O}(\log^* n)$ time.

In the third component, we deal with the remaining edges that cannot be put into the maximal conflict-free edge set. Each remaining edge conflicts with at least one edge in current TPS graph G_{TPS}. For each conflict, we first try to separate it through calculating new underlying paths. Thanks to Lemma 7, we know that the length of an underlying path p_e is bounded by $8(k+1)^2$. Hence, if a conflict is separable, we can eliminate it within $\mathcal{O}(\Delta^{16(k+1)^2})$ time by enumerating all possible underlying paths. For a remaining edge $e = (u, v)$ entangled in non-separable conflicts, our method locally tests whether there exists a path p in G_{TPS} such that p connects u and v and the length of p is bounded by a constant μ. We fix the length of an alternate path to be a small constant 3, i.e., $\mu = 3$. These tests can be finished in constant time. If all the remaining edges are disposed, our algorithm finishes and outputs a planar graph with small distortion. In the example in Fig. 3, the remaining two edges $(14, 10)$ and $(23, 15)$ can both be replaced with path of two hops in the TPS, such as paths 23-8-15 or 23-1-15 for $(23, 15)$. Our algorithm outputs the final TPS, shown in Fig. 3i. For comparison, CDM graph is also calculated in this example, shown in Fig. 3j. As a result, our algorithm can be finished in polynomial time. Further, when the maximal node degree Δ is bounded by a constant, the time complexity of our algorithm reduces to $\mathcal{O}(\log^* n)$.

5 Discussion

We discuss more details in this design. We deal with complicated conflict resolution, and construct the planar embedding of TPS.

5.1 Additional Optimizations

We here present a simple solution to isolated nonseparable conflicts. Let edges e_1 and e_2 be a pair of conflicting edges in RWG. The conflict between e_1 and e_2 is *isolated*, if they do not conflict with any other edges in RWG. Given an isolated nonseparable conflict between edges $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$, we locally modify the original RWG as follows: we aggregate four tiles associated with landmarks u_1, v_1, u_2, and v_2 into a super tile. A new landmark is selected in this super tile such that its hop distance to other nodes in the super tile is at most $k' = 4k + 2$. Such modification thus removes isolated non-separable conflicts from RWG and TPS. A TPS remains a subgraph of RWG. Clearly, if all nonseparable conflicts are isolated and eliminated by this method, a $(2\mu k' + 1, 2k')$-TPS is obtained.

We propose an additional technique to ensure the connectivity of G_{TPS}. Clearly, when G is connected, the RWG G_R built from G is connected. We further guarantee that G_{TPS} is connected if G_R is connected. In particular, each node in G_{TPS} tries to flood a color in G_{TPS}. The color is represented by its node ID or a randomly generated number. During flooding, the smallest color value suppresses the others. When this flooding procedure finishes, G_{TPS} is dominated by one color value if G_{TPS} is connected. Otherwise, each connected component of G_{TPS} is identified by a different color. We further consider nodes in G_R with the same color as a single virtual node, and conceptually obtain a overlay graph G' on top of G_R. One node of G' corresponds to one connected component of G_{TPS}. If G_{TPS} is disconnected, G' will contain more than one node. We then build a spanning tree T on G'. One edge in T may correspond to multiple edges in G_R. For each edge $e = (u, v)$ in T, we select one edge in $E(G_R) \backslash E(G_{TPS})$ to represent it, and thus obtain a set of edges in G_R, denoted by E_T. We add all these edges E_T into G_{TPS}. The updated G_{TPS} is guaranteed to be a connected planar graph. For the distributed implementation of these operations, the key issue is to ensure that only one edge in $E(G_R) \setminus E(G_{TPS})$ is selected to bridge two connected components of G_{TPS}, out of a possibly large number of edges connecting the two components. This can be achieved by another round of message flooding. After previous message flooding, nodes in one connected component share a common color, different with the colors of other components. Let l be a node in G_{TPS}. If the color of at least one of l's neighbors in G_R is different with that of l, we say l is a *border node* in G_{TPS}. Border nodes witness the adjacency between connected components. Each border uses the composite value of its color and ID as its new color and floods messages. Messages with the smallest color in alphabetical order still suppresses the others. When this flooding procedure finishes, only one edge in $E(G_R) \setminus E(G_{TPS})$ is selected to connect two components of G_{TPS}.

As a last remark, this work utilizes a uniform sampling (kMIS) of the network as landmarks to build an RWG, which makes risky edges in RWG usually appear scatteredly and sparsely. Our extensive simulations also verify that the technique for isolated nonseparable conflicts is practically effective enough to resolve all remaining edges in RWG. Conceptually, however, it is possible to construct an RWG instance artificially such that nonseparable risky edges are hinged into a large component of network scale. It is indeed impossible to solve such ill cases using only local connectivity information. If forced to find planar spanner for such RWG cases, we have to collect connectivity information of network scale and partially utilize centralized computation, which is too costly for large networks. It is more feasible and efficient to perform our method on those ill cases as follows: we first recognize and locate the network regions where large hinged crossing components happen. In those regions, we randomly regenerate the landmarks and rebuild a new network partition. The updated RWG can mostly make the ill scenarios disappear with high probability. We can thus planarize the whole network in a divide-and-conquer fashion.

5.2 Construct Planar Embedding of TPS

It is important and useful to construct a planar drawing for TPS. For example, face structure of a planar graph can be obtained through a planar embedding. Face information is explicitly required in many applications of network planarization, such as geometric routing. We can use the method discussed in previous work [13] to construct a planar embedding of TPS, because TPS has a better connectivity than CDM and contains a lot of triangles. First, we identify a triangle in TPS graph as landmarks, e.g., in Fig. 3k, nodes in triangle $\langle 1, 11, 13\rangle$ are selected as landmarks. We initially assign coordinates of an equilateral triangle to the vertices of landmarks. Then we fix the landmarks and iteratively place every vertex into the center of gravity of its neighbors in a distributed manner [23]. When this process comes to an equilibrium state, as illustrated in Fig. 3k, we obtain a planar straight-line drawings of TPS. The planar embedding assigns each vertex in TPS graph a virtual coordinate. Hence, each face of TPS is calculated in a distributed manner as face routing operations in geometric routing protocols, such as GFG [1], GPSR [2], and GOAFR [3]. Further, we can look for the longest face cycle in this embedding in terms of hop numbers, denoted by the bold lines in Fig. 3k. The longest face cycle typically corresponds to the face cycle of the outer boundary in the network. If we place this longest face cycle on a unit circle (or regular polygon) and repeat the above embedding procedure, a different planar embedding can be obtained. In such embedding the edges are usually of more proper length than those in the initial embedding, as shown in Fig. 3l.

Note that this rubber-banding embedding technique itself does not guarantee to find a unique planar embedding of TPS. A graph has a unique embedding if and only if it is a subdivision of a triconnected planar graph, as described in

Theorem 1.1 in the literature [24]. The problem of computing all possible planar embeddings of planar graphs is well studied in the field of planar graph drawing, and can be solved in polynomial time [25].

6 Correctness of Algorithm

This section analyzes the correctness and performance of our algorithm. We first prove planarity of TPS, and then discuss the connectivity of TPS. The main results are shown in Theorems 1, 2, and 3.

We present Lemmas 1, 2, 3, and 4 before proving Theorem 1. Given a quasi-UDG H with an embedding ε, we can use the embedding ε to draw the vertices and edges of H in the plane, and thus obtain an image of H in it. We define this image as the *embedding map* of H, denoted by $\mathcal{M}^{\varepsilon}(H)$, or simply $\mathcal{M}(H)$. For example, the union of all lines in Fig. 3e can be viewed as an embedding map of all the underlying paths. We further define the *shadow* of H as follows: The shadow $\mathcal{S}^{\varepsilon}_{\ell_0}(H)$ of H, or simply $\mathcal{S}(H)$, is the union of the embedding map $\mathcal{M}(H)$ and the solid polygons that are surrounded by cycles not greater than ℓ_0 in H. In this definition of graphs' shadow all cycles of size ℓ, $\ell \leq \ell_0 = 4$, are filled in and become solid polygons [20]. For conciseness from now on, we refer to $1/\sqrt{2}$-quasi-UDG as quasi-UDG unless otherwise specified. Note that geometric realizations of quasi-UDGs are used in the later proofs, but they cannot be obtained explicitly, when only connectivity information is given. These proofs indeed exploit the properties of combinatorial quasi-UDGs that are independent of specific embeddings. It is sufficient for our proofs that those geometric realizations conceptually exist.

Lemma 1. *Given two noncontiguous paths p_1 and p_2, their embedding maps $\mathcal{M}(p_1)$ and $\mathcal{M}(p_2)$ do not share any one point.*

Proof. This directly follows from the link-crossing property of quasi-UDGs. □

Lemma 2. *Given a vertex v and ℓ-size cycle C in G, $\ell \leq \ell_0$, point $\mathcal{M}(v)$ does not fall into the shadow $\mathcal{S}(C)$ of cycle C, if $d_G(v, C) \geq 2$.*

Proof. Let v' be any vertex in cycle C and $\|\mathcal{M}(v)\mathcal{M}(v')\|$ denote the euclidean distance between points $\mathcal{M}(v)$ and $\mathcal{M}(v)$. If $\mathcal{M}(v)$ is located in the interior of $\mathcal{S}(C)$ in any valid realization of G, it is not difficult to verify that $\|\mathcal{M}(v)\mathcal{M}(v')\| \leq 1/(2sin(\pi/\ell_0)) \leq 1/\sqrt{2}$. Hence, we have $d_G(v, C) = 1$ when G is $1/\sqrt{2}$-quasi-UDG, which leads to contradiction. □

Lemma 3. *The embedding maps of two separable paths either do not intersect or intersect even number of times.*

Proof. Let p_1 and p_2 be two separable paths. According to the definitions of cycle-homotopy paths and separable-conflict testing, we know the following facts: first, there exist two paths p_1' and p_2' in G such that $p_1 \simeq p_1'$, $p_2 \simeq p_2'$, and p_1' and p_2' are not contiguous; second, the concatenation $\widehat{p_1 p_1'}$ (or $\widehat{p_2 p_2'}$) admits a cycle partition $\mathcal{C}_1$ (or $\mathcal{C}_2$) in G such that the length of each cycle in $\mathcal{C}_1$ (or $\mathcal{C}_2$) is ℓ_0 at most. Let S_1 be the union of the following plane regions: shadow $\mathcal{S}(C)$ of every cycle $C \in \mathcal{C}_1$, images $\mathcal{M}(p_1)$ and $\mathcal{M}(p_1')$. S_2 is defined in the same way as S_1. We know that the two endpoints of p_1 (or p_2) do not fall into the shadow S_2 (or S_1), due to Lemma 2 and the fact that a landmark and its one hop neighbors belong to the same tile in the RWG when the tile size $k \geq 2$. Hence, if the embedding maps $\mathcal{M}(p_1)$ and $\mathcal{M}(p_2)$ intersect, $\mathcal{M}(p_1)$ (or $\mathcal{M}(p_2)$) can be smoothly deformed to $\mathcal{M}(p_1')$ (or $\mathcal{M}(p_2')$) without leaving the region S_1 (or S_2, respectively). As a result, $\mathcal{M}(p_1)$ and $\mathcal{M}(p_2)$ either do not intersect or intersect even number of times. □

Lemma 4. *Given an RWG G_R with an underlying representation (L, P) and a subgraph H of G_R, H is planar if any two edges in H are conflict-free or the conflict between them is separable with respect to the underlying representation (L, P).*

Proof. We first explore the underlying representation and draw H in the plane. Given an edge e in H, we use p_e to denote the underlying path of e. We utilize the embedding maps of underlying nodes and paths to draw the vertices and edges in H, respectively. For each edge $e = (u, v)$ in H, we find a simple (nonself-intersecting) polygonal curve in the embedding map $M(p_e)$ that connects points $M(u)$ and $M(v)$. We use $D(e)$ to denote this polygonal curve drawn for e. Let e_1 and e_2 be any two edges in H that do not share any endpoint. If e_1 and e_2 are conflict-free, $D(e_1)$ and $D(e_2)$ do not intersect due to Lemma 1. If the conflict between e_1 and e_2 is separable, $D(e_1)$ and $D(e_2)$ either do not intersect or intersect even number of times according to Lemma 3. Thus, we construct a drawing for H such that the drawn paths for every pair of non-adjacent edges in H either do not intersect or intersect in even times. Hence, H is planar, following Theorem A in literature [26], which shows even edge crossings does not destroy the planarity of a graph. □

Theorem 1. *Given a combinatorial ρ-quasi-UDG G with $\rho \leq 1/\sqrt{2}$, a TPS G_{TPS} constructed from G by our algorithm is a connected planar graph.*

Proof. In this design, we first extract a maximal conflict-free graph from RWG as the initial TPS. We further extend the TPS by performing separable conflict resolution on the remaining edges in RWG. Planarity of TPS is preserved in these steps due to Lemma 4. Moreover, other complicated conflict resolution mechanisms discussed in Section 5.1 also preserve the planarity of TPS. The connectivity of TPS is ensured by an additional technique presented in Section 5.1. □

We next show the spanner property of restricted witness graph in Lemma 5.

Lemma 5. *Given an RWG G_R constructed from G with parameter k, a $(2k+1, 2k)$-spanner of G can be built from G_R.*

Proof. We construct a spanning subgraph G' of G such that G' is a $(2k+1, 2k)$-spanner. The construction goes as follows: for each landmark vertex v in G_R, we build a shortest path tree T_v in the tile rooted at v. For each edge $e = (u, v)$ of G_R, we select one shortest path connecting

u and v in G, denoted by p_e. G' is set to be the subgraph of G that is induced by all the edges in the shortest path trees, denoted by $E_T = \bigcup_{v \in V(G_R)} E(T_v)$, and edges in the shortest path, denoted by $E_p = \bigcup_{e \in E(G_R)} E(p_e)$, that is, $E(G') = E_T \bigcup E_p$. For any two nodes u and v in G, let p be one shortest path in G' that connects u and v and passes through m landmarks. Clearly, $|p| \leq (2k+1) \cdot (m-1) + 2k \leq (2k+1) \cdot d_G(u,v) + 2k$, as the shortest path connecting u and v in G travels at most $d_G(u,v)+1$ different tiles. □

Generally, it is easy to show that when we relax the restrictions on the selection of landmarks and permit any node in a tile serve as the landmark of the tile, a $(2\delta+\lambda, 2\delta)$-spanner of G can be constructed from a (δ, λ)-restricted witness graph of G.

If all the edges left are disposed through performing conflict edge resolution, our constructed planar graph is a $(2\mu k+1, 2k)$-TPS, as shown in Theorem 2.

Theorem 2. *A $(2\mu k+1, 2k)$-TPS is obtained after successfully performing conflict edge resolution.*

Proof. This directly follows from Lemma 5 and Theorem 1. □

We next compare TPS with CDM in Theorem 3. More concretely, Funke and Milosavljevic [13] indicate that a combinatorial Delaunay map faithfully reflects the topology of the network, and prove a CDM has good connectivity when built from large tiles. We show that TPS is at least a supergraph of CDM. On some instances it is strictly better, as illustrated by the examples in Figs. 1, 2, and 3. Our experiments show that TPS outperforms CDM on many practical instances as well.

Theorem 3. *Given a combinatorial ρ-quasi-UDG G with $\rho \leq 1/\sqrt{2}$, when TPS and CDM are constructed by using the same landmarks and tiles in G, TPS is a supergraph of CDM.*

Proof. Let (l, l') be any edge in a CDM graph. According to the rules of CDM construction, we know that there exists a path in G that connects landmarks l and l' and consists of a sequence of nodes in tile l followed by a sequence of nodes in tile l', meanwhile, one-hop neighbors of the path contains only nodes in tile l or tile l'. Any possible conflict between two CDM edges is clearly separable. Hence, given any possible network instances, a CDM graph will be subgraph of a TPS constructed by this design. □

Finally, we present Lemmas 6 and 7, which are required in the analysis of complexity of algorithm.

Lemma 6. *Given a combinatorial ρ-quasi-UDG G with $\rho \leq 1/\sqrt{2}$ and an RWG G_R constructed from G by our algorithm, the maximum node degree of G_R is bounded by a constant.*

Proof. This directly follows the packing property of quasi-UDG, that is, the maximum independent nodes that can be packed in a unit of area is bounded by a constant. □

Lemma 7. *Given a combinatorial ρ-quasi-UDG G with $\rho \leq 1/\sqrt{2}$, an RWG G_R constructed from G by our algorithm with tile parameter k, and any edge e in G_R, the length of any underlying path of e is bounded by $8(k+1)^2$.*

Proof. Let $l, l' \in V(G_R)$ be two landmarks and $e = (l, l') \in E(G_R)$. Let $V_l \subseteq V(G)$ be all the nodes affiliated with tile l. For any node $v \in V_l$, we know that $d_G(l,v) \leq k$ according to the construction of RWG. The euclidean distance between nodes v and l is, thus, not greater than k. Let n_0 be the maximum number of independent nodes that we can place in the big disk with radius $k+1$ under $1/\sqrt{2}$-quasi-UDG model. Clearly, $n_0 < (\pi(k+1)^2)/(\pi(1/\sqrt{2})^2) = 2(k+1)^2$. Two tiles l and l' thus contain at most $4(k+1)^2$ independent nodes. Hence, the chordless underlying path p_e of e contains at most $8(k+1)^2-1$ nodes. □

7 Evaluation

We conduct extensive simulations to evaluate the effectiveness of this approach. We compare this design with two state-of-the-art approaches: CDM graph-based planarization proposed by Funke and Milosavljevic [13], denoted by CDM, and robust planarization proposed by Zhang et al. [14], denoted by ZJC. They are recognized as the most important connectivity-based and distributed planarization methods in wireless ad hoc and sensor networks.

7.1 Qualitative Evaluation

In this set of simulations, we compare TPS with CDM and ZJC qualitatively. We uniformly deploy nodes in regions with different shapes. By default, the networks are generated under 0.8-quasi-UDG model.

We first present some results to visually demonstrate the quality of TPS, and then examine the large distortions of CDM and ZJC. In particular, we vary the average node degrees and network shapes to show that our approach is robust to network density and shape, as shown in Figs. 4a, 4b, 4c, 4d, 4e, 4f, 4g, and 4h. We can see that TPS outputs dense well-connected graphs in all those network configurations. We compare TPS with CDM in the same networks. We increase the density of kMIS landmarks and reduce tile's sizes by changing the parameter k, from 5 to 2. Figs. 4i, 4j, 4k, 4l, 4m, 4n, 4o, and 4p show a set of results in a "G"-shape network. Compared with TPS, CDM is rather sparse and has a large distortion. CDM is disconnected and contains many connected branches, when parameter k becomes smaller and kMIS landmarks form a dense sampling of the network.

We further perform ZJC in the networks that have the same shapes as Figs. 4e, 4f, 4g, and 4h. The results of ZJC are shown in Figs. 4q, 4r, 4s, and 4t. We can see that two shortest path trees built by ZJC may cover only a small portion of the network, which causes large distortions among nodes that are not covered by the trees. By varying many different shapes, we find ZJC mainly works in networks with regular square-like shapes. When the network regions have feature-rich outer boundary or inner holes and thus cannot be split into two triangular subregions, the two shortest path trees built by ZJC cannot cover the whole network well, which could result in arbitrarily large distortion in the planar graph extracted by ZJC.

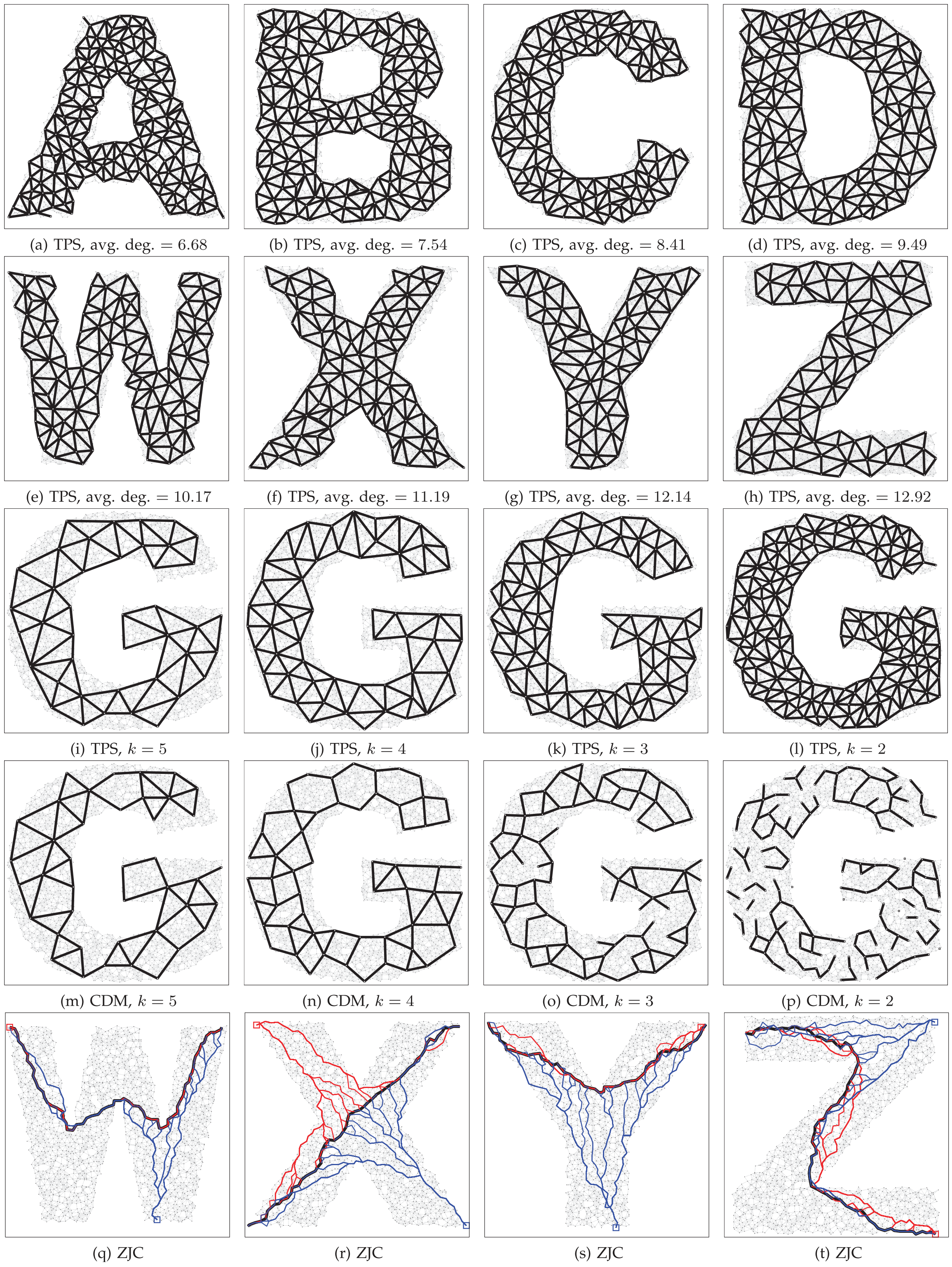

(a) TPS, avg. deg. = 6.68
(b) TPS, avg. deg. = 7.54
(c) TPS, avg. deg. = 8.41
(d) TPS, avg. deg. = 9.49
(e) TPS, avg. deg. = 10.17
(f) TPS, avg. deg. = 11.19
(g) TPS, avg. deg. = 12.14
(h) TPS, avg. deg. = 12.92
(i) TPS, $k = 5$
(j) TPS, $k = 4$
(k) TPS, $k = 3$
(l) TPS, $k = 2$
(m) CDM, $k = 5$
(n) CDM, $k = 4$
(o) CDM, $k = 3$
(p) CDM, $k = 2$
(q) ZJC
(r) ZJC
(s) ZJC
(t) ZJC

Fig. 4. Qualitative evaluation on TPS, CDM, and ZJC. (a)-(h) show TPS results in networks with various shapes and average node degrees, tile radius $k = 2$. (i)-(p) compare TPS and CDM, varying tile radius k from 5 to 2. (q)-(t) are the results of ZJC, and each figure plots two trees rooted at two square nodes and the *base path* [14] denoted by bold line.

Due to the page limit, we skip many results on CDM and ZJC in network fields of various different shapes. Generally, we have the following observations: TPS and CDM are independent from the network shape since they only use local connectivity information, while ZJC needs global connectivity and probes to network shapes. Hence, we will mainly compare TPS with CDM in the following quantitative results.

7.2 Quantitative Results

In this set of simulations, we quantitatively examine the distortion of TPS in practical networks. We deploy

TABLE 1
Planarization Results of TPS

	Stretch for landmark nodes					Stretch for nodes in the whole network					Comparison with CDM					
k	α	t_{max}	σ	t_{avg}	σ	α, β	t_{max}	σ	t_{avg}	σ	*branch*	σ	*degree*	σ	*ratio*	σ
2	13	1.931	0.035	1.194	0.033	13,4	7.621	0.231	1.406	0.104	36.533	2.271	0.783	0.018	4.413	0.102
3	19	1.830	0.046	1.171	0.033	19,6	8.115	0.511	1.468	0.137	2.533	0.427	1.264	0.027	2.495	0.055
4	25	1.693	0.034	1.154	0.034	25,8	8.603	1.104	1.527	0.168	1.167	0.126	1.536	0.025	1.848	0.032
5	31	1.608	0.029	1.139	0.035	31,10	9.278	1.114	1.601	0.203	1.033	0.061	1.683	0.033	1.495	0.032
6	37	1.518	0.050	1.120	0.034	37,12	9.987	1.139	1.659	0.225	1.033	0.061	1.739	0.041	1.277	0.026

2,500 nodes in a square area by uniformly random distribution. We use 0.8-quasi-UDG to build connectivity graphs with an average node degree around 11. We vary the densities of kMIS landmarks, k from 2 to 6. Under each configuration, our simulation takes 100 runs with random network generation and reports the average. The results are shown in Table 1.

We first measure the multiplicative stretches between landmark nodes. In particular, a TPS G_{TPS} is transformed into an edge-weighted graph as follows: for each edge $e = (u, v)$ in G_{TPS}, the weight of e is equal to the hop number of the shortest underlying path of e. The multiplicative stretch between any two nodes u and v in G_{TPS} is the ratio of their distance in G_{TPS} to distance in connectivity graph G, that is, $d_{G_{TPS}}(u, v)/d_G(u, v)$. Both the worst and average stretches are computed. The worst stretch is the maximum stretch between any two nodes, and the average stretch calculates the average of all stretches between any two nodes. t_{max} and t_{avg} are the worst and average stretch, respectively. σ is the standard deviation. Our method finishes normally in all these simulations, and achieves a small stretch factor. Accordingly (α, β)-spanner is also guaranteed due to the successful implementation of our method. These theoretical values of α and β are calculated according to Theorem 2. From Table 1, we can see TPS performs well in practical networks. Especially when $k = 2$, the worst and average stretch factors are less than 2 and 1.2 respectively, much better than the theoretical results provided in Theorem 2.

We next check the multiplicative stretches between any two nodes in the whole network. We need to construct a spanning graph G' of G from a TPS. G' is built in the same way as discussed in the proof of Lemma 5, and multiplicative stretch between any two nodes u and v in G is $d_{G'}(u, v)/d_G(u, v)$. We can see the worst and average stretch factors are still greatly better than the theoretical bounds in Table 1. Further, a TPS with smaller tile's size k provides a better multiplicative stretch for the network. This is consistent with our intuition that a TPS with small tiles achieves a dense sampling of the network and produces a spanner with a small addictive stretch, thus can better reflect real network distances.

We finally compare TPS with CDM. We find that there always exist some instances in each network configuration where CDM graph is disconnected, which makes it infeasible to calculate the average stretch factor for CDM. Instead, we have to utilize other metric to measure the quality of CDM. In Table 1, we use *branch*, *degree*, and *ratio* to denote the number of connected branches, vertex degree of CDM, and the ratio of the number of edges in TPS to that in CDM, respectively. From Table 1, we can see that TPS has much better connectivity than CDM, especially when tile's sizes decrease.

8 Conclusions

As a crucial issue in wireless ad hoc and sensor networks, network planarization is previously addressed either under ideal assumptions, or in relaxed models while not providing any guarantee on the quality in terms of connectivity or distortion. We present a practical method to perform topological planar simplification on networks, and take the first attempt towards extracting a provably planar topology from the network in a fine-grained and location-free manner. The simulation results show that our method always produces planar graphs with small stretch factors, which significantly outperforms the state-of-the-art approaches.

Acknowledgments

The authors are grateful for a variety of valuable comments from the anonymous reviewers. Dezun Dong was supported in part by the NSFC under grants No. 61170261 and No. 61170284. Xiangke Liao was supported in part by the National High Technology Research and Development Program of China (863 Program) under grant No. 2012AA01A301. Yunhao Liu was supported in part by the NSFC Distinguished Young Scholars Program under Grant No. 61125202. The research of Xiang-Yang Li was partially supported by US National Science Foundation (NSF) CNS-0832120, NSF CNS-1035894, the National Natural Science Foundation of China under Grant No. 61170216, and the program for Zhejiang Provincial Key Innovative Research Team program for Zhejiang Provincial Overseas High-Level Talents (One-hundred Talents Program). Zhengbin Pang was supported in part by the NSFC under grants No. 61272482 and No. 61202488.

References

[1] P. Bose, P. Morin, I. Stojmenović, and J. Urrutia, "Routing with Guaranteed Delivery in Ad Hoc Wireless Networks," *Proc. ACM Third Int'l Workshop Discrete Algorithms and Methods for Mobile Computing and Comm. (DIALM),* 1999.

[2] B. Karp and H. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," *Proc. ACM MobiCom,* 2000.

[3] F. Kuhn, R. Wattenhofer, and A. Zollinger, "Worst-Case Optimal and Average-Case Efficient Geometric Ad-Hoc Routing," *Proc. ACM MobiHoc,* 2003.

[4] S. Funke and N. Milosavljevic, "Guaranteed-Delivery Geographic Routing under Uncertain Node Locations," *Proc. IEEE INFOCOM,* 2007.

[5] D. Moore, J. Leonard, D. Rus, and S. Teller, "Robust Distributed Network Localization with Noisy Range Measurements," *Proc. ACM Second Int'l Conf. Embedded Networked Sensor Systems (SenSys)*, 2004.

[6] Y. Wang, J. Gao, and J.S. Mitchell, "Boundary Recognition in Sensor Networks by Topological Methods," *Proc. ACM MobiCom*, 2006.

[7] J. Gao, L. Guibas, J. Hershberger, L. Zhang, and A. Zhu, "Geometric Spanner for Routing in Mobile Networks," *Proc. ACM MobiHoc*, 2001.

[8] X.-Y. Li, G. Calinescu, P.-J. Wan, and Y. Wang, "Localized Delaunay Triangulation with Application in Ad Hoc Wireless Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 14, no. 10, pp. 1035-1047, Oct. 2003.

[9] Y. Kim, R. Govindan, B. Karp, and S. Shenker, "Geographic Routing Made Practical," *Proc. Second Conf. Symp. Networked Systems Design and Implementation (NSDI)*, 2005.

[10] J. Chen, A. Jiang, I. Kanj, G. Xia, and F. Zhang, "Separability and Topology Control of Quasi Unit Disk Graphs," *Proc. IEEE INFOCOM*, 2007.

[11] Y. Wang, "Topology Control for Wireless Sensor Networks," *Wireless Sensor Networks and Applications*, chapter 5, pp. 113-147, Springer, 2008.

[12] G. Mao, B. Fidan, and B. Anderson, "Wireless Sensor Network Localization Techniques," *Computer Networks*, vol. 51, no. 10, pp. 2529-2553, 2007.

[13] S. Funke and N. Milosavljevic, "Network Sketching or: How Much Geometry Hides in Connectivity?-Part II," *Proc. 18th Ann. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 2007.

[14] F. Zhang, A. Jiang, and J. Chen, "Robust Planarization of Unlocalized Wireless Sensor Networks," *Proc. IEEE INFOCOM*, 2008.

[15] D. Dong, M. Li, Y. Liu, X.-Y. Li, and X. Liao, "Topological Detection on Wormholes in Wireless Ad Hoc and Sensor Networks," *IEEE/ACM Trans. Networking*, vol. 19, no. 6, pp. 1787-1796, Dec. 2011.

[16] Q. Fang, J. Gao, L. Guibas, V. de Silva, and L. Zhang, "GLIDER: Gradient Landmark-Based Distributed Routing for Sensor Networks," *Proc. IEEE INFOCOM*, 2005.

[17] A. Nguyen, N. Milosavljevic, Q. Fang, J. Gao, and L.J. Guibas, "Landmark Selection and Greedy Landmark-Descent Routing for Sensor Networks" *Proc. IEEE INFOCOM*, 2007.

[18] D. Dong, X. Liao, Y. Liu, C. Shen, and X. Wang, "Edge Self-Monitoring for Wireless Sensor Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 22, no. 3, pp. 514-527, Mar. 2011.

[19] S. Baswana, T. Kavitha, K. Mehlhorn, and S. Pettie, "New Constructions of (α, β)-Spanners and Purely Additive Spanners," *Proc. Ann. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 2005.

[20] D. Dong, X. Liao, K. Liu, Y. Liu, and W. Xu, "Distributed Coverage in Wireless Ad Hoc and Sensor Networks by Topological Graph Approaches," *IEEE Trans. Computers*, vol. 61, no. 10, pp. 1417-1428, Oct. 2012.

[21] V. de Silva and G. Carlsson, "Topological Estimation Using Witness Complexes," *Proc. Symp. Point-Based Graphics*, 2004.

[22] J. Schneider and R. Wattenhofer, "A Log-Star Distributed Maximal Independent Set Algorithm for Growth-Bounded Graphs," *Proc. ACM 27th ACM Symp. Principles of Distributed Computing (PODC)*, 2008.

[23] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica, "Geographic Routing without Location Information," *Proc. ACM MobiCom*, 2003.

[24] T. Nishizeki and N. Chiba, *Planar Graphs: Theory and Algorithms.* Elsevier Science, 1988.

[25] N. Chiba, T. Nishizeki, S. Abe, and T. Ozawa, "A Linear Algorithm for Embedding Planar Graphs Using PQ-Trees," *J. Computer and System Sciences*, vol. 30, no. 1, pp. 54-76, 1985.

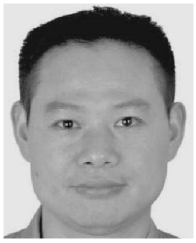
[26] J. Pach and G. Toth, "Which Crossing Number Is it Anyway?" *J. Combinatorial Theory, Series B*, vol. 80, no. 2, pp. 225-246, 2000.

Dezun Dong received the BS, MS, and PhD degrees from the National University of Defense Technology (NUDT), China, in 2002, 2004, and 2010, respectively. He was a visiting scholar in the Computer Science and Engineering Department of the Hong Kong University of Science and Technology from November 2008 to May 2010. Currently, he is an assistant professor in the School of Computer, NUDT, China. His research interests include wireless networks, distributed computing, and high-performance computer systems. He is a member of the IEEE.

Xiangke Liao received the BS and MS degrees in computer science from Tsinghua University and National University of Defense Technology (NUDT), China, in 1985 and 1988, respectively. He is now a professor and the dean at the School of Computer, NUDT, China. His research interests include parallel and distributed computing, high-performance computer systems, operating system, and networked embedded system.

Yunhao Liu received the BS degree in automation from Tsinghua University, Beijing, China, in 1995, and the MS and PhD degrees in computer science and engineering from Michigan State University in 2003 and 2004, respectively. Being a member of Tsinghua National Lab for Information Science and Technology, he holds a Tsinghua EMC chair professorship. He is the director of the Key Laboratory for Information System Security, Ministry of Education, and a professor in the School of Software, Tsinghua University. He is also a faculty member in the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. His research interests include pervasive computing, peer-to-peer computing, and sensor networks. He is a senior member of the IEEE.

Xiang-Yang Li received the bachelor's degree in business management and the BEng degree in computer science from Tsinghua University, P.R. China, in 1995, and the MS and PhD degrees in computer science from the University of Illinois at Urbana-Champaign in 2000 and 2001, respectively. He has been an associate professor since 2006 and an assistant professor of computer science at the Illinois Institute of Technology from 2000 to 2006. His research interests include cyber physical systems, wireless sensor networks, social networks, and security. He is an editor of the *IEEE Transactions on Parallel and Distributed Systems*; the *IEEE Transactions on Mobile Computing*; *Networks*; and *Ad Hoc and Sensor Wireless Networks*; and was a guest editor of special issues for several journals, including the *IEEE Journal on Selected Areas in Communications* and *ACM Mobile Networks and Applications*. He coedited several books, including the *Encyclopedia of Algorithms*, published in 2008. In 2008, he published a monograph, *Wireless Ad Hoc and Sensor Networks: Theory and Applications* (Cambridge University Press). He is a senior member of the IEEE and ACM.



Zhengbin Pang received the BS, MS, and PhD degrees in computer science from the National University of Defense Technology (NUDT), China, in 1994, 1997, and 2007, respectively. He is a professor in the School of Computer, NUDT, China. His research interests include parallel and distributed computing, and high-performance computer systems.