A Localized Planarization Algorithm for Realistic Wireless Networks

Emi Mathews  
University of Paderborn, Germany  
Email: emi@hni.upb.de

Hannes Frey  
University of Paderborn, Germany  
Email: hannes.frey@uni-paderborn.de

Abstract—Planar graph routing works provably correct if the underlying network graph is connected and planar. Typically, wireless networks modeled as 2D graphs, are not planar and planar graph routing applied on such unprocessed network graphs may fail. Planarizing a given connected graph by removing intersecting links might be impossible if the outcome still needs to be a connected subgraph. It becomes even more difficult with distributed planarization techniques, where each node is allowed to use only the information about its local neighborhood. Furthermore, it is getting complicated if the nodes’ assigned positions do not reflect the exact physical location. With or without exact location information, the outcome might be disconnected, nonplanar, or both of it.

With all these unsolvable problems, the question arises how to apply planar graph routing in a realistic network setting? Fortunately, wireless network graphs bear one property which distinguishes them from arbitrary graphs: due to limited communication range, network links cannot become arbitrarily long. In this work we exploit this locality property to build a new localized planarization algorithm, which is location fault tolerant and which produces planar connected graphs in most cases in realistic wireless models. We evaluate our algorithm using the Log Normal Shadowing model and show that our algorithm always produces planar connected graphs in all simulations even when large location errors are present.

Keywords—localized planarization; topology control; k-hop clustering; overlay graph; location fault tolerance; realistic wireless networks

I. INTRODUCTION

Efficient routing in large-scale ad-hoc networks such as wireless sensor networks (WSN) and wireless mesh networks (WMN) is a challenging task due to frequent topology changes, bandwidth constrained links and resource constrained nodes. With the availability of small, inexpensive, and low-power Global Positioning System receivers [1] or other localization systems [2], routing based on geographic information gained significant attention. With geographic routing, a forwarding node can decide the next hop node by just considering the geographic location of its neighbor nodes. Any changes in the network topology, like link or node failures, require routing information updates only at the nodes located close to that change. Traditional topology-based routing in contrast requires global routing information updates in such cases.

The basic geographic routing approach works by greedily forwarding the messages to nodes which minimize a local forwarding metric, such as distance to the destination. For all greedy routing variants, forwarding might end up at a node which is the best compared to its neighbor nodes. However, from these nodes the messages cannot be forwarded further in greedy mode according to the forwarding metric, though a path from source to destination may exist.

So far, FACE routing, originally described in [3], is the only known localized single path approach to recover from such greedy forwarding failures. FACE routing uses localized right/left hand graph traversal, to find a path around the boundaries of the void regions. FACE routing guarantees message delivery if it is applied on a planar embedding of the communication network. A planar embedding is a graph representation in a plane where edges intersect only at their end points. In general, a wireless network topology is not a planar embedding. Hence, a planarization step prior to routing is required.

Planarization methods typically construct planar subgraphs by removing links from the original network topology. Distributed planarization is difficult for real wireless networks [4]. Existing distributed planarization protocols [5], [6] designed for arbitrary networks are non-local; which means that neighborhood information beyond a constant hop might be needed for planarization. Existing localized planarization techniques [3], [7], [8], i.e., methods where each node is required to know only its one (or two) hop neighbors, work well for unit disk graph [9]. They do not work correctly in realistic wireless networks, which in most cases do not obey the unit disk graph property.

In addition, localized planarization methods assume that exact node position information is at hand. However, in practice, geographic location information is not exact; because of the unavailability or limitation of resources or because of errors in the localization techniques. Location inaccuracies degrade the performance of localized planarization algorithms. They cause incorrect edge removal during planarization and may disconnect the planar subgraph. Even small errors can lead to incorrect routing decisions with noticeable performance degradation [10], [11].

Thus, we have the following dilemma: there are topology control schemes which transform arbitrary 2D graphs (including localization faults) into a topology where FACE routing is always successful. However, these approaches are not localized. On the other hand, all known localized approaches cannot be correctly applied on arbitrary 2D
graphs and do not work when location errors are present. In this paper, we solve this dilemma utilizing a basic structural property of realistic wireless networks; i.e. the links cannot be arbitrarily long. We describe a localized planarization algorithm for realistic wireless networks which is location fault tolerant and produces planar graphs in most cases, using this property.

The remainder of this paper is structured as follows. Section II describes the planarization algorithm. In Section III, we explain the wireless models used in this work. Next, in Section IV we present a simulation based performance evaluation and analysis of the proposed algorithm. Related approaches known from the literature are then discussed in Section V. Finally, Section VI summarizes the main results of this work and provides an outlook on possible future research.

II. TOPOLOGICAL CLUSTER-BASED PLANARIZATION ALGORITHM

We assume ad-hoc wireless networks as a graph \( G = (V, E) \) embedded in the two dimensional Euclidean space. The graph has a finite set of vertices \( V \) that corresponds to the nodes in the network. Each node knows its 2D position which need not be the exact one. The set of edges \( E \) corresponds to the wireless links between the nodes. We consider dynamic networks with frequent link changes but stationary nodes. This is typical for WSN or WMN.

The algorithm is based on the observation that Euclidean edge lengths in ad-hoc wireless network graphs can not be arbitrarily long, as seen in Figure 1a. The key idea of our approach is that, at a local scale even though the network graph might contain many intersecting links, at a larger scale those intersections disappear. More precisely, when we aggregate nodes into clusters, the limited edge length assures that the cluster interconnections do not intersect in most cases.

Our planarization algorithm starts with topology-based clustering to aggregate nodes into clusters. The topology-based clustering is a distributed process which uses only local information exchange. Clustering creates a logical hierarchy in the network with dedicated cluster heads handling all inter-cluster routing decisions. An overlay graph is constructed with the cluster heads constituting its vertices. Links joining the neighboring cluster heads constitute the edges in the overlay graph. Figure 1b shows an overlay graph created by topological clustering for illustration.

The overlay graphs may contain a few intersections and may not be planar. Hence a localized cross link detection and repair algorithm is executed afterwards, which finally makes the overlay graph planar in most cases.

The details of the planarization process stages are described next. The pseudo-code showing their implementations can be found in the Appendix A.

A. Clustering

The first stage of the planarization algorithm is cluster formation. In this stage the nodes are grouped into clusters based on any multi-hop clustering algorithm. We have developed a clustering algorithm called k-hop clustering based on the CONID (connectivity ID) [12], where \( k \) is the cluster depth or hop count. Our algorithm differs from [12] in the selection process of cluster head and cluster member nodes. This makes it more robust against node location errors, as we see later in the analysis. The k-hop clustering algorithm creates clusters with cluster heads that are separated at least \( k \)-hops apart. It uses connectivity as the primary key in clustering decisions, where connectivity is defined as:

**Definition 1:** The connectivity of a node \( u \), \( \text{Con}(u) = |S_u| \), where \( S_u = \{v_1, v_2, \ldots\} \) such that \( \forall \) nodes \( v_i \in S_u \), a path \( |p(u, v)\) \leq k \) from \( u \) to \( v \), exists and nodes along \( |p(u, v)| \) do not belong to another cluster. For \( k = 1 \), the connectivity is equivalent to the node degree.

The k-hop clustering algorithm works as follows. All nodes periodically check their status to determine whether they are clustered or not. If there exists any node which does not belong to any cluster, it is triggered to create a new cluster or join an existing cluster. If the triggered node is an isolated node, i.e. without any neighbors, it changes its status to clustered and creates a new cluster with the current node as its only member node. All other triggered nodes create a k-hop neighbor list \( S \) and calculate their connectivity as given in definition 1.

The node with higher connectivity is preferred to nodes with lower ones. The nodes which have the highest connectivity among their k-hop neighbors become winner nodes. If two nodes \( u \) and \( v \) have the same connectivity and their connectivities are maximum in the set \( S_{u \cup \{u\}} \) and \( S_{v \cup \{v\}} \) respectively, then their node ids are compared to make the decision. The one with the lower id then wins.

Prior to the cluster creation, a check is done to determine whether the connectivity of the node is above a threshold size, which is usually \( k \). If connectivity is not above the threshold size, the node finds a cluster in which the majority
of its neighbors are members, and joins that cluster. The node then requests all nodes in $S$ to join this cluster. This connectivity-threshold step prevents the formation of clusters with very few members in the network. Though prevention of numerous small clusters is optional from the point of view of planarization, but it helps to make the algorithm robust against node location errors. Due to the joining of new nodes, the cluster depths are increased at most by the threshold size. Usually it is increased only by a small fraction of the threshold size. Choosing small threshold values keeps this increase limited. In our experiments, the threshold is chosen to be the cluster depth $k$.

B. Overlay graph

The next process in the planarization algorithm is the overlay graph construction. An overlay network could be constructed using the winner nodes which initiated the cluster creation; but to make the overlay vertices location fault tolerant, we create virtual nodes at the geometric center of the clusters. Virtual node average out location errors in the cluster. The position of the virtual nodes are calculated as:

$$P_v(x, y) = \sum_{i=1}^{n} \frac{P_i}{n}, \quad (1)$$

where $n$ is the number of nodes in the cluster and $P_i$ is the position of the $i^{th}$ member of the cluster. These nodes constitute the virtual cluster heads ($VCH$) of the clusters. In practice, the winner node or a node which is close to the centroid of the cluster serves the functions of $VCH$.

The node id of $VCH$ is the actual id of the node which is serving it. The cluster is also represented by this id. After creating the $VCH$, the information about the $VCH$ and the node serving it, is passed to all members of the cluster.

A virtual overlay graph $OG = (V', E')$ is created with the virtual cluster heads as its vertices. For a network graph $G = (V, E)$, the set of vertices of the overlay graph $V' = \bigcup_{i} \{VCH(v_i)\}$ for all $v_i \in V$, where $VCH(v_i)$ is the virtual cluster head of node $v_i$. The set of edges of the overlay graph $E' = \bigcup\{e_i\}$, where an edge $e_i$ exists between $VCH_i$ of cluster $C_i$ and $VCH_j$ of cluster $C_j$, if $\exists w \in C_i$ and $w \in C_j$ such that, an edge $(v, w) \in E$ in $G$. For example, the virtual edge between $VCH_1$ and $VCH_2$ shown in Figure 1b exists, as there is a link between nodes $v$ and $w$ in the network graph.

For constructing the virtual links, each $VCH$ sends queries to the boundary nodes of its cluster. If any boundary node has a link to another node with a different $VCH$, a virtual link is created between these cluster heads.

C. Cross link detection and repair

The last stage of the planarization algorithm is the local cross link detection and repair (CLDR) algorithm. The overlay graphs are mostly planar especially when the $k$ values are large, but when $k$ is small and network density is very high, they may not be planar. The CLDR makes the overlay graphs planar in most cases.

The CLDR begins with cross link detection. For cross link detection, all nodes $v'_i \in V'$ in the overlay graph $OG(V', E')$, collect their 2-hop neighbor information. The 2-hop neighbors of $v'_i$ are those nodes which are at-most 2-hops away from $v'_i$, when traversed over the virtual links $e' \in E'$. With this information, each node checks if there exist any locally detectable crossing links. An intersection is detected locally from a node $u$, if a link $uv$ towards its one-hop neighbor $v$ is intersected by another link $wx$ from $u$’s another one-hop neighbor $w$ such that $x \neq v$ and $x \neq u$.

If such cross links are detected, the overlay graph is not planar. Hence the repair step of the CLDR is executed. It removes one of the cross links, if the link removal is safe. A link is considered safe to be removed, when its removal does not disconnect the network. If the link removal is not safe, the links are retained and the overlay graph remains nonplanar. The repair step uses the following rules for detecting safe links and then performs the link removal operation.

- If there exists a path from $u$ to $v$, other than the direct link $uv$, with path length at-most 2-hops, then remove the link $uv$.
- Else, if there exists a path from $u$ to $x$, not through the node $w$, with path length at-most 2-hops, then remove the link $ux$.

When an intersecting link is removed the node lying on the other side of this link get immediate notification from the node that executes removal. The removed links are not considered anymore during the cross link detection or repair phase. This ensures that the network does not get disconnected when other nodes in the network perform similar local repair operations on crossing links.

III. MODELING AND SIMULATION

A. Wireless Model

We evaluate our planarization algorithm using the Log Normal Shadowing Model (LNS) [13]. LNS takes the presence of obstacles into its account. The path loss, the ratio of radiated power to the received power $\frac{P_{tx}}{P_{rec(d)}}$, at a distance $d$ expressed in decibel is:

$$PL(d)[dB] = PL(d_0)[dB] + 10\gamma \log_{10} \frac{d}{d_0} + X_\sigma[dB], \quad (2)$$

where $\gamma$ is the path loss exponent, $X_\sigma$ is a zero-mean Gaussian random variable with variance $\sigma^2$, and $PL(d_0)$ is the reference path loss at a reference distance $d_0$. The random component is added to the signal attenuation to reflect the signal strength variations caused by shadowing, which is time invariant in our simulation.

The path loss exponent $\gamma$ varies between 2 (free-space) and 6 (obstructed in-buildings) whereas, variance $\sigma^2$ varies between 3.7 and 12.8 [14], [15]. In indoor environments,
for a frequency of 2.45 GHz, $\gamma = 2$ for line of sight (LOS) and $\gamma = 3.5$ for non-line of sight (NLOS) [16]. We choose $\gamma = 3.25$, slightly less than the NLOS value and $\sigma = 2.5$ for most of the experiments, but we also vary them to study their effects on planarity. The reference path loss $PL(d_0)$, for a reference distance $d_0 = 1 m$ varies between $-50$ and $-30 \text{ dBm}$ [14] and we set $PL(d_0)$ to $-40 \text{ dBm}$. The path loss can be calculated from equation 2 using these values.

The received signal strength indicator (RSSI) at a distance $d$, for a given transmitter output power $P_{tx}[\text{dBm}]$, is:

$$RSSI(d)[\text{dBm}] = P_{tx}[dBm] - PL(d)[dB],$$

(3)

The transmission power levels of IEEE 802.11 often vary between $-15 \text{ dBm}$ to $15 \text{ dBm}$ [17]. We set the $P_{tx}$ to $15 \text{ dBm}$. Two nodes separated at a distance $d$ are considered as neighbors when the $RSSI(d)[\text{dBm}]$ is greater than the receiver’s sensitivity, $RxS$. The receiver sensitivity varies with data transmission rates. E.g. IEEE 802.11g’s $RxS$ is $-88 \text{ dBm}$ at $6\text{ Mbps}$ and $-66 \text{ dBm}$ at $54\text{ Mbps}$ [17]. We set the $RxS$ to $-80 \text{ dBm}$ for our experiments.

Besides LNS, we consider the most widely used wireless network models such as Unit Disk Graph (UDG) [9] and Quasi Unit Disk Graph (d-UQDG) [18], as well in our simulations. In UDG, all nodes have the same transmission range $R$. We set $R$ to $50$ units in our tests. A more general model is the d-UQDG which contains all edges shorter than radio range $r$ and no edges longer than another range $R$, where $d = \frac{R}{r}$. The existence of an edge for node distances between $r$ and $R$ is not specified. We set $d$ to 0.5, with $r = 25$ and $R = 50$ units. The performance of our algorithm at this $d$ value, which is less than $\frac{1}{2}$ is specially interesting, as existing localized planarization algorithms cannot guarantee planarity at such low values [19].

**B. Simulation setup**

We use the ShoX network simulator [20] for network creation and algorithm execution. We create different networks by varying the parameters like field size and average node degree. Average node degree $D$ is defined as:

$$D = \frac{\pi r^2}{A} \times N,$$

(4)

where $r$ is the transmission radius, $A$ is the area of the field, and $N$ is the total number of nodes in the network.

In the remainder of the paper we use a parameter $\rho = \frac{D}{N}$ to represent the node degree. In most of the experiments, we use the field size $500 \times 500$, with $N$ varying from 100 to 500. To test the scalability of the algorithm, we also consider a field size of $2500 \times 2500$, with $N$ varying from 2500 to 12500. The tests conducted on the former field are evaluated on 100 different network configurations while on the latter field, are evaluated only on 10 different network configurations due to the enormous time consumption of each test.

A graph planarity test is conducted to check the planarity of the overlay graph. It classifies the graphs into planar and nonplanar groups and reports the total number of intersecting links in nonplanar cases.

**IV. PERFORMANCE EVALUATION AND ANALYSIS**

To analyze the performance of the planarization algorithm, we vary parameters such as hop count $k$ used in k-hop clustering and node degree $\rho$. We record the percentage of nonplanar graphs in the LNS modeled networks. Table I shows the result of the planarization algorithm on two field sizes, $500 \times 500$ and $2500 \times 2500$, with $\rho$ varying from 2 to 5. Hop count $k$ varies from 1 to 6 in the first field and 2 to 12 in the second field. The column $k.a$ indicates the percentage of nonplanar graphs before applying CLDR step and the column $k.b$ indicates the percentage after applying CLDR.

Table I: Performance of the planarization algorithm

<table>
<thead>
<tr>
<th>Field Size</th>
<th>500 x 500</th>
<th>2500 x 2500</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho$</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>a b</td>
<td>a b</td>
</tr>
<tr>
<td>2</td>
<td>100 0</td>
<td>60 0</td>
</tr>
<tr>
<td>3</td>
<td>100 0</td>
<td>100 0</td>
</tr>
<tr>
<td>4</td>
<td>100 0</td>
<td>100 0</td>
</tr>
<tr>
<td>5</td>
<td>100 0</td>
<td>100 0</td>
</tr>
</tbody>
</table>

Table I: Performance of the planarization algorithm

Figure 2: Effect of cluster depth on planarity
most important: overlay graphs are always planar in all test cases when CLDR is applied. Figure 2 shows the effect of hop count on planarity more clearly, where the percentage of nonplanar graphs without CLDR is plotted at a confidence level of 95% for three different network models. The field size used in these experiments is $500 \times 500$ and the node degree ($\rho$) is 3. It shows that, for smaller $k$ values we get fewer planar overlay graphs. With increasing $k$ value, more graphs become planar. When $k$ is increased to values comparable to 25% of the network diameter, most of the overlay graphs are planar.

The experiments conducted on the field size $2500 \times 2500$ also affirm that planarity increases with the hop count. At higher $k$ values like 10 or 12, the majority of the test cases are planar. If we increase $k$ further to values comparable to the network diameter, the probability of getting planar overlay graphs is very high as the total number of clusters in the network at large $k$ values is very small. This decreases the probability of having intersecting links.

To obtain planar overlay graphs with low $k$ values, the CLDR step is needed. Columns $b$ of table I show that with the CLDR step, all graphs are planar irrespective of the average node degree. It is also interesting to note that for small hop counts such as $k=1$, the overlay graphs are planar in all simulation runs. However, we may need $k > 1$ in certain situations, as discussed in Section IV-C and Section IV-D.

A. Performance of existing planarization algorithms

Graph algorithms such as Relative Neighborhood Graph (RNG), Gabriel Graph (GG) create planar sub-graphs locally from the full network graph using distributed algorithms [3], [7]. Most of the existing geographic routing protocols such as GPSR [7] and GOAFR [21] use the GG planarization method for planar-graph-based void handling. Delaunay Triangulation (DT)-based graphs have shorter recovery paths than RNG or GG. A localized construction of DT-based spanner is feasible [8]. These algorithms provably yield connected planar graphs in connected UDG networks. We now evaluate their performance in LNS and QUDG modeled networks.

Figure 3 shows the results of the experiments at a confidence level of 95% on the field size $500 \times 500$, when $\rho$ is varied from 1 to 5. The figure shows that DT, GG and RNG perform very badly in planarizing realistic wireless network graphs. For $\rho > 2$, less than 10% of the GG subgraphs are planar and for the same $\rho$ value, none of the DT subgraphs are planar. The planarization algorithm we proposed, always created planar graphs, irrespective of node degrees in the LNS and QUDG modeled networks.

B. Effect of model parameters on planarity

The experiments discussed in the above section are based on fixed values for path loss exponent $\gamma$ and variance $\sigma^2$ of LNS model. In this section, we analyze the effect of these parameters on the planarity of the graphs.

To study the effect of LNS variance, we vary $\sigma$ fixing the $\gamma$ value. Figure 4 shows the result of the planarization algorithms at a confidence level of 95% on field size $500 \times 500$ with $\rho \approx 3$ and $\gamma = 3.25$. Empirical studies show that $\sigma^2$ varies between 3.7 and 12.8 [14], [15]. Hence, we vary $\sigma$ from 0 to 4 ($\sigma^2$ from 0 to 16) in our experiments. The results of DT, GG and RNG planarization algorithms show that the variance has significant impact on the planarity of the graphs. Planarity decreases with the increase in variance. As variance increases, nodes that are closer may not be connected any more, but those far apart may get connected. Such link irregularities lead to intersections which cannot be repaired without causing disconnection.

The results of our planarization algorithm with CLDR, show that the proposed algorithm is robust against the $\sigma$ variations. The $k=1$ planarization algorithm has a few nonplanar cases, but only at $\sigma$ values outside the normal range. The $k=2$ planarization algorithm always creates...
planar graphs even for extreme variances. This is because larger $k$ values subside local link irregularities better.

In a study on the effect of path loss exponent on planarity by varying $\gamma$ fixing $\sigma$, we could not find any direct impact of $\gamma$ on planarity.

C. Effect of localization errors

To study the effect of location inaccuracy on the performance of the planarization algorithm, location errors are added to the true node positions. We used the Gaussian error model described in [22] for our analysis. There the mean value is set to 0 and standard deviation $\sigma_{err}$ is varied from 0 to 20% of the radio range. In our experiments, we vary $\sigma_{err}$ from 0% to 150% of the radio range, to find out the effect of extreme $\sigma_{err}$ values. The radio range used in the LNS model is calculated using the equation 5, assuming $\sigma = 0$.

\[
\text{radio range} = 10^{(P_t - P_L(d_0) - Rxs)/(10 \times \text{gamma})}
\]  \( (5) \)

Figure 5 shows the results of the experiments at a confidence level of 95% on the field size 500 x 500 with $\rho \approx 3$ in the LNS networks. DT, GG and RNG produce nonplanar graphs in most of the simulations when there is a small location error; but our algorithm produces planar graphs in all test cases for $\sigma_{err} \leq 100\%$ of the radio range. In addition to nonplanarity, location errors also cause disconnection in some networks when DT, GG or RNG is used, whereas our algorithm does not disconnect any connected networks in our experiments. When the location error increases, especially when $\sigma_{err} \geq 100\%$ of the radio ranges, our algorithm with $k = 1$ produces a few nonplanar graphs. The $k = 1 \ast$ plot shows the results of our algorithm without the connectivity-threshold step, in the clustering stage. It shows that clustering with the connectivity-threshold step reduces nonplanarity.

As soon as $k$ is increased to 2, non-planarity disappears, even at extreme location inaccuracies. Hence, in networks with extremely high node localization errors, we propose using larger cluster depths. In those networks with reasonable location accuracy, i.e. $\sigma_{err} < 100\%$ of the radio range, one hop clustering is sufficient to create planar overlay graphs.

The location fault tolerance of the planarization algorithm is due to the fact that edge removal occurs at the VCH level rather than the node level. Moreover, the position of the VCH is robust against individual location errors, especially when the cluster depth increases as the errors are averaged out. Hence the virtual link removal is less vulnerable to location errors than the physical links.

V. Related Work

There has been considerable amount of research in planarizing network graphs. However, the question arises whether a localized planar graph construction in realistic networks is possible or not.

A. Localized planarization in restricted wireless model

Graph algorithms such as Gabriel Graph (GG), Relative Neighborhood Graph (RNG) and Delaunay Triangulation (DT) are commonly used to planarize network graphs [14]. In [3], [7], a localized construction of GG and RNG is presented. DT-based graphs have shorter recovery paths than RNG or GG and a localized construction of DT-based spanner is presented in [8], [23]. However, several messages need to be exchanged for constructing such planar graphs. Reactive planar spanner construction using angle-based and Delaunay-based direct planarization techniques is presented in [24]. All algorithms provably yield a connected planar graph as long as the network obeys the unit disk graph property. If the network graph violates the UDG assumption, the planarization results may contain network partitions, unidirectional links and intersecting links. Moreover, localization errors also affect the planarization process as discussed in section I. An approach presented in [11] addresses the incorrect edge removal during planarization due to localization errors in UDG models. However, this approach do not function well in realistic wireless models.

In Quasi Unit Disk Graphs, [19] gives an approach to produce planar graphs that are connected by adding virtual edges, which are essentially tunnels through multiple existing links, prior to the Gabriel Graph planarization step. The GG planarization succeeds on this augmented graph without partitioning it.

An alternate idea to planarize graphs is to use virtual nodes instead of the virtual links [25]. To obtain planarity, each edge intersection is replaced with a virtual node and a real node serves as a proxy for the virtual node. This approach does not work in arbitrary graphs as well. It requires that intersections can be detected locally.

Proactive construction of grid structures for planar void handling has been discussed in [26]-[31]. In these works, geographic clusters are created by partitioning the plane.
by an infinite mesh of regular polygons such as square or hexagonal grids, with the cluster diameter set to the transmission range of a node. Nodes are aggregated to such clusters and the overlay graph is created from the adjacent clusters. Due to the UDG assumption, packet delivery cannot be guaranteed in realistic wireless models.

**B. Non-local algorithms for arbitrary network models**

The only known technique which works for arbitrary graphs is to produce a safe routable subgraph by probing each link and removing some of the crossed edges, as in the Cross-Link Detection Protocol (CLDP) [5]. FACE routing does not fail in this subgraph. An alternative to this technique is Lazy Cross-Link Removal [6] which does not proactively remove the cross links but applies CLDP when loops are found. CLDP based protocols are not local in the sense that the probe messages may have to travel between nodes which are more than constant hops from each other.

A proactive construction that creates hull trees for void handling is presented in [32]. The associated routing protocol Greedy Distributed Spanning Tree Routing, switches to routing on a spanning tree instead of routing on planar faces.

Reactive void handling algorithms activate path discovery strategies only when the packet gets stuck at void nodes. Different distributed algorithms exist to identify holes and discover hole-surrounding paths. Geometric properties of voids are used in BOUNDHOLE [33], partial flooding at void nodes is used in [34], [35] and link-reversal-based destination-oriented DAG construction is used in [36], [37]. However, none of these protocols are local.

Landmark-based or anchor based geographic routing is presented in [38], [39] where a global map of the anchors are discovered and maintained by the source or obtained from the global information such as maps for greedy-packet-forwarding along anchored paths. In [40] a planar landmark graph is created with local information which eliminates the need for keeping landmark maps.

**VI. Conclusion and future work**

We proposed a localized planarization algorithm that creates planar graphs in realistic wireless models and which is location fault tolerant. The planarization algorithm creates an overlay graph by topology-based clustering. Using a cross link detection and repair algorithm, the intersections in the overlay graph are removed locally. Simulation studies show that the existing localized planarization algorithms perform badly in planarizing networks with realistic wireless models, whereas the algorithm we proposed planarized all networks used in the simulations.

Node location errors worsen the performance of the existing localized planarization algorithms. With small location errors, $\sigma_{err} = 5\%$ of the radio range, these algorithms failed to produce any planar graphs in the simulated networks. Location errors cause disconnections too in some of the simulated networks. The planarization algorithm we proposed does not disconnect any network. Moreover, it is location fault tolerant, which is achieved by increasing the cluster depth. In networks with reasonable location accuracy, one hop clustering algorithm is sufficient to create planar overlay graphs. The additional overhead of our algorithm due to the clustering step is significantly low in this case. By increasing hop count to 2, the algorithm could achieve location fault tolerance in networks with extreme location inaccuracies as well.

Geographic routing algorithms such as GPSR and GOAFR could be applied at a macroscopic level, using the planar overlay graph provided by our planarization algorithm. We are currently working on the specification of a routing protocol, where we plan to use two different routing modes, an inter-cluster mode and an intra-cluster mode. In the intra-cluster routing mode, nodes could use greedy forwarding to send packets to their destinations. For inter-cluster routing, nodes could send packets to those $VCH$s in their neighborhood that minimize a forwarding metric. At a void $VCH$, planar graph routing along the virtual faces of the overlay graph could be employed. For sending packets to a $VCH$ outside one node’s cluster, simple strategies such as partial flooding (within the cluster) could be used to reach its cluster boundary node(s). From the boundary node, packets are forwarded to any boundary node of the adjacent $VCH$ and then forwarded further greedily to the $VCH$.

Besides the routing protocol, we plan to make theoretical analysis about the planarity of overlay graphs created from the topological clustering in simpler network models such as UDG. We also plan to investigate the potential improvement that could be achieved with our algorithm in reducing the overhead of current location services by keeping virtual cluster heads instead of real nodes. This needs to be validated with more quantitative results. We conclude that the geometric properties of the overlay graph produced from topological clustering is an area worth exploring more.

**References**


Algorithm A.1 k-hop Clustering

1: procedure cluster-all-nodes
2: while \( \forall \ Node n \in V : n.\text{status} \neq \text{CLUSTERED} \) do
3: \( \text{if} \ neighbors(n) = \emptyset \) then
4: \( n.\text{status} = \text{CLUSTERED} \)
5: end if
6: trigger-cluster-node \( (n) \)
7: end while
8: end procedure

Algorithm A.2 Overlay graph

1: procedure Overlay (Cluster \( C_i \))
2: Create virtual cluster head \( \text{VCH} \)
3: \( \text{VCH}.\text{id} = \text{Winner}.\text{id} \)
4: \( j=1 \)
5: for all \( \text{Node } n_j \in C_i \) do
6: \( \text{Pos}_{\text{VCH}} = \text{Pos}_{\text{VCH}(j-1)} + \text{Pos}_{n_j} \)
7: for all \( \text{Node } n \in \text{neighbors}(n_j) \) do
8: \( \text{if} \ n.\text{clusterId} \neq \text{VCH}.\text{id} \) then
9: \( \text{neighbors}(\text{VCH}).\text{add}(n,\text{VCH}) \)
10: end if
11: end for
12: end for
13: end procedure

Algorithm A.3 CLDR

1: procedure CLDR (OverlayNode \( u \))
2: for all \( v \in \text{neighbors}(u) \) do
3: for all \( w \in \text{neighbors}(u) \setminus \{v\} \) do
4: \( \text{for all } x \in \text{neighbors}(w) \setminus \{u, v\} \) do
5: \( \text{if} uv \text{ intersects } wx \) then
6: \( \text{if} \ \exists y \in N(u) \setminus \{v\} \& u - y, y - v \in E \) then
7: remove\( (uv) \)
8: \( \text{else if} \ (\exists y \in N(u) \setminus \{w\} \cup \{u\} \& uy, yx \in E) \) then
9: remove\( (wx) \)
10: end if
11: end if
12: end for
13: end for
14: end for
15: end procedure

APPENDIX


