Maximal Cliques in Unit Disk Graphs: Polynomial Approximation

Rajarshi Gupta[1], Jean Walrand[1], Olivier Goldschmidt[2]
[1] Department of Electrical Engineering and Computer Science
University of California, Berkeley, CA 94720, USA
{guptar, wlr}@eecs.berkeley.edu
[2] OPNET Technologies Inc, 2006 Delaware Street Berkeley, CA 94709, USA
ogoldschmidt@opnet.com

Abstract

We consider the problem of generating all maximal cliques in an unit disk graph. General algorithms to find all maximal cliques are exponential, so we rely on a polynomial approximation. Our algorithm makes use of certain key geographic structures of these graphs. For each edge, we limit the set of vertices that may form cliques with this as the longest edge. We then consider several characteristic shapes determined by that edge, and prove that all cliques having this as the longest edge, are included in one of the sets of vertices contained in these shapes. Our algorithm works in $O(m\Delta^2)$ time and generates $O(m\Delta)$ cliques, where m is the number of edges in the graph and Δ is its maximum degree. We also provide a modified version of the algorithm which improves the performance in many cases, albeit without affecting the worst case running time.

Keywords: Cliques, Unit Disk Graphs, Ad-Hoc Networks.

1 Introduction

We address the problem of finding all maximal cliques in an Unit Disk Graph (UDG) [1]. An unit disk graph $G = (V, E)$ is defined on a finite set of points V in the plane. Two vertices $u, v \in V$ are connected by an edge $uv \in E$ if and only if their Euclidean distance is less than or equal to 1. An induced subgraph in G that is a complete graph is called a *clique*. A *maximal clique* of G is one that it is not contained in any other clique.

UDGs have recently attracted a lot of importance due to the advent of wireless ad-hoc networks. Modelling of ad-hoc networks requires the understanding of interference between neighboring nodes and links. Two nodes/links are often modelled as interfering when they lie within an interference range of each other, making the underlying graph a UDG [2]. UDGs also appear in channel assignment problems in broadcast and cellular networks (e.g. [1]). Cliques in such networks are of importance since only a single vertex in a clique may be active at once – leading to considerations of capacity and quality of service.

In this paper, we attempt to find all maximal cliques in a UDG, in polynomial time. We begin by describing in Sec. 2 the background of the problem, and the related work in the field. Our approximation algorithm and its rationale is presented in Sec. 3. We complete the paper with a discussion of simulation results in Sec. 4, and a conclusion in Sec. 5.

2 Background and Related Work

We cite a sampling of papers in the ad-hoc networking arena that utilize the concept of cliques for scheduling, routing and QoS purposes. In [3], the author uses these ideas for optimizing traffic flows, while the authors of [2] and [4] use cliques to derive necessary and sufficient bounds on the maximum capacity of an ad-hoc network. In [5], the authors propose a clique-based pricing approach to optimize resource allocation in an ad-hoc network. All these ideas require the computation of maximal cliques, preferably in a distributed manner.

Cliques have been studied in great detail in the area of graph theory. Algorithms to generate maximal cliques from a graph were first introduced by Harary and Ross [6] in 1957. During the 1960s and 1970s, the Bierstone Algorithm was developed [7, 8], and was further refined in [9]. All of these algorithms work on a general graph and output maximal cliques. The number of maximal cliques in a graph is typically exponential, as such algorithms are consequently (e.g. [10]). While the above algorithms also work for UDGs, our specific applications often require a quicker approximation approach.

It is tempting to hope that the special structure of UDG's caps the number of maximal cliques in a UDG. In fact, the number of cliques could be exponential in n, the number of vertices in the graph, as we have shown.

Theorem 1 *[11] The total number of cliques in a UDG grows exponentially with n in the worst case.*

Proof 1 *We illustrate this by the following example. Assume $n = 2p$. Draw a circle of diameter $1+\epsilon$, where $0 < \epsilon << 1$. Place the $2p$ nodes uniformly on the edge of the circle and label them clockwise from 1 to $2p$. If ϵ is small enough, we have edges from any vertex i to all other vertices except the diametrically opposite vertex $(i+p)$ mod $2p$. Thus we have constructed a complete p-partite graph with its vertex set being a pair of diametrically opposite nodes i.e. $\{i, i+p\},\ i = 1, \ldots p$. The selection of one vertex from each of the p sets will form a maximal clique. Clearly we have a total of $2^p = 2^{n/2}$ maximal cliques.*

This pessimistic result led us to look at suitable approximations, that would enable a polynomial-time algorithm. The essence of the approximation is to use slightly 'super-maximal' cliques. When the number of cliques grows large, the approximation algorithm generates the *union* of several nearby cliques as a single super-maximal clique.

In [12], we have earlier presented an approximation algorithm to find all maximal cliques in ad-hoc networks. We recognized two key features about the geographic nature of ad-hoc networks. First, two nodes that are part of a clique must be within an interference range $\omega = 1$ of each other. Second, if a group of nodes form a clique, then the maximum distance between any pair of them must be 1. The heuristic approximation uses a small disk of diameter 1 (i.e. radius = $1/2$) to scan a larger disk of radius 1 around a node. Alternatively, the scanning disk is used to scan the entire region in which the nodes are placed. Each position of the scanning disk generates a clique. The generated set of cliques is then shrunk to result in the approximate set of maximal cliques around the node.

There are however a few problems with the scanning disk approach presented in [12]. The foremost of which is that the running time of the algorithm depends on the step size and the size of the field. Further, the scanning disk of diameter 1 fails to catch all cliques, e.g., three nodes located at the corners of an equilateral triangle of side 1. We would like to address these issues as we move beyond the realm of ad-hoc networks, to more generic UDGs.

3 Algorithm

We order the edges in G in decreasing order of length. Then, for each edge uv, we find all maximal cliques with uv as the longest edge. This method will generate all maximal cliques in G, together with some extra cliques that may be subsets of other larger maximal cliques. The set of cliques generated may be processed at a later stage to prune the non-maximal cliques.

3.1 Important Shapes in UDG

We begin by observing several important geometric structures determined by an edge, and their characteristics w.r.t. maximal cliques in UDGs. We consider three shapes in particular – we call these the *football*, the *disk* and the *curved triangle*.

3.1.1 Football

Given an edge uv, let d_{uv} be the Euclidean distance between u and v. Obviously, $d_{uv} \leq 1$, since our graph is an UDG. We draw two circles of radius d_{uv} centered at u and v. Denote the set of vertices in the football-shaped (American Football, not Soccer) intersection of the two circles as F_{uv}, shown in Fig. 1(a). Then we prove the following.

Theorem 2 *[11] A clique whose maximum edge is uv must be contained in F_{uv}.*

Proof 2 *Suppose the clique with uv as the longest edge contains a vertex $w \in V \setminus F_{uv}$. Then, either $d_{uw} > d_{uv}$ or $d_{vw} > d_{uv}$. In either case, uv is not the longest edge. (Contradiction)*

However, F_{uv} in itself is not a maximal clique as defined. An example is shown in Fig. 1(a), where nodes $i, j \in F_{uv}$. Yet $d_{ij} > d_{uv}$ and hence not part of a maximal clique with uv as the longest edge. In fact, we could even have $d_{ij} > 1$, in which case, $ij \notin E$. Thus F_{uv} is a superset of the requisite maximal cliques.

By simple geometry, the height of the football F_{uv} is $d_{uv}\sqrt{3}$. A special situation occurs when $d_{uv} \leq 1/\sqrt{3}$. In this case, the height of the football is ≤ 1; so every vertex in F_{uv} is connected to each other. Then there is in fact a single clique F_{uv}, which includes all maximal cliques with uv as the longest edge.

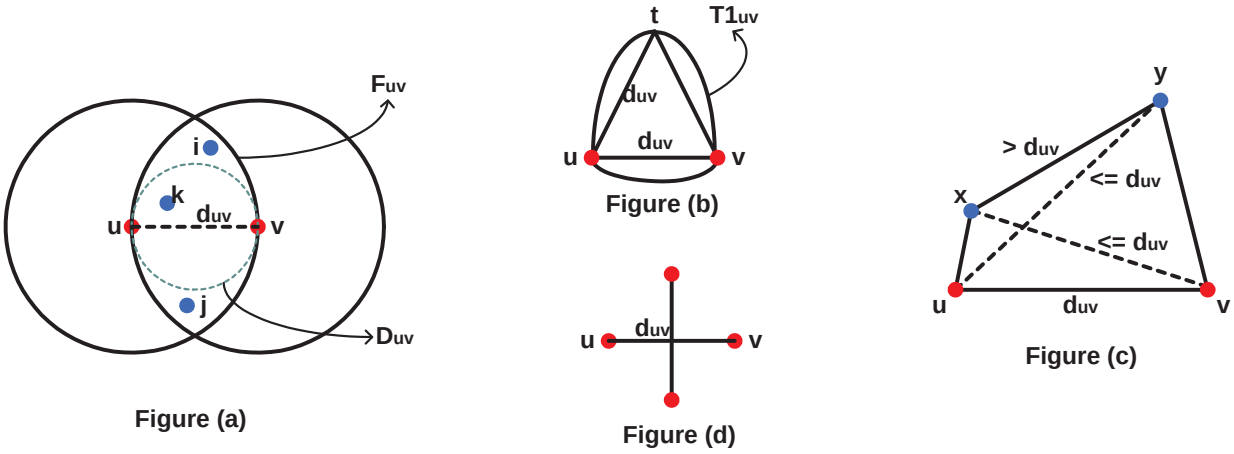


Figure 1: Characteristic Shapes in UDG: (a) Football F_{uv} (b) Curved Top Triangle $T1_{uv}$ (c) Proof that $T1_{uv}$ is a clique (d) Counter-example showing $T1_{uv}$ and $T2_{uv}$ not enough to cover all cliques

3.1.2 Disk

The next shape of interest is the disk D_{uv} which is the set of all vertices in the circle with uv as its diameter, also shown in Fig. 1(a). Clearly, the maximum distance between any two nodes in D_{uv} is d_{uv}. Thus D_{uv} is a clique with uv as its longest edge. Note however that D_{uv} may not always be maximal, e.g., in Fig. 1(a), $D_{uv} = \{u, v, k\}$, yet the clique $\{u, v, k, i\}$ contains D_{uv}.

3.1.3 Curved Triangle

Finally, we look at the curved triangle formed by u, v, and t – the upper intersection point of the two circles of radius d_{uv} centered at u and v, as denoted in Fig. 1(b). The three points form an equilateral triangle. We expand this area by drawing curves with radius d_{uv} from u to v centered at t and so on. All vertices in the resulting top curved triangle is denoted by $T1_{uv}$. A similar bottom curved triangle $T2_{uv}$ may also be drawn. The vertices in these form cliques, as we prove below.

Theorem 3 *The vertices in $T1_{uv}$ or $T2_{uv}$ form a clique with uv as the longest edge.*

Proof 3 *We show that any two vertices in $T1_{uv}$ are separated by a distance $\leq d_{uv} \leq 1$. Then, $T1_{uv}$ forms a clique with uv as the longest edge. Suppose $x, y \in T1_{uv}$ and $d_{xy} > d_{uv}$. We choose two points out of the equidistant points u, v, t such that both x and y lie on the same side as the edge connecting these two points. $Walog$, we choose u and v. The resulting quadrilateral formed by joining them to x and y is shown in Fig. 1(c). Note that all edges except xy are $\leq d_{uv}$.*

$$\text{In } \triangle uxv,\ uv \geq ux, vx. \text{ Then } \angle uxv \geq \angle xuv \geq \angle xuy. \tag{1}$$

$$\text{But in } \triangle uxy,\ xy > ux, uy. \text{ Then } \angle xuy > \angle uxy \geq uxv. \tag{2}$$

Thus we have a contradiction.

However, the curved triangle also does not necessarily cover all cliques. For instance, the four vertices in a cross shown in Fig. 1(d) form a clique, but can not be covered by any curved triangle as defined.

To summarize, we have outlined three characteristic shapes determined by the edge uv:

- Football F_{uv} is a superset of all maximal cliques with uv as the longest edge.
- Disk D_{uv} is a clique with uv as the longest edge, but may not be maximal.
- Curved Triangles $T1_{uv}$ and $T2_{uv}$ are cliques with uv as the longest edge. But these too may not be maximal.

If every node in F_{uv} is contained in any of D_{uv}, $T1_{uv}$ or $T2_{uv}$, then we have found the single maximal clique with uv as the longest edge. Since that may not always happen, we need an alternative method.

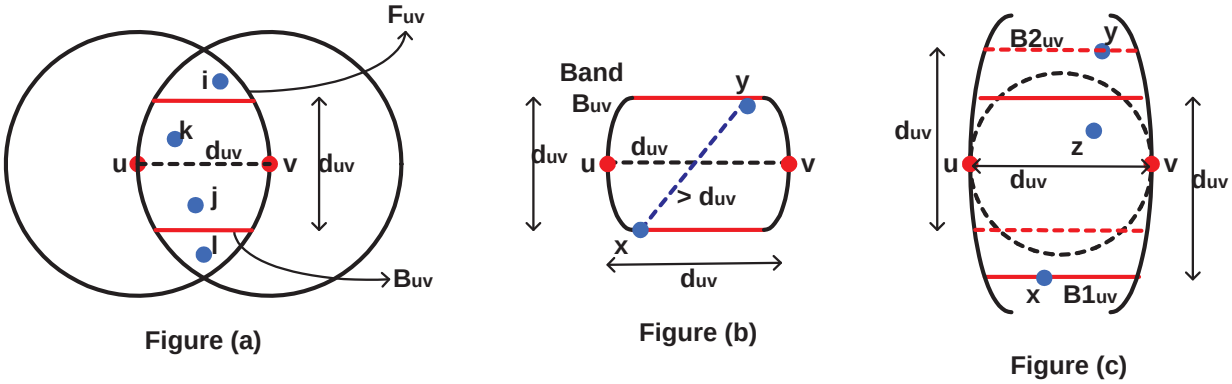


Figure 2: Moving band in football: (a) Band B_{uv} in Football (b) B_{uv} may include some extra vertices (c) Positioning bands with boundaries on vertices in F_{uv}

3.2 Moving Band in Football

Consider the football shaped area as before, and look at a band of height d_{uv} within this, as shown in Fig. 2(a). Let B_{uv} be the set of nodes within the band. Since the height of the band is greater than half of the height $d_{uv}\sqrt{3}$ of the football, any such band always contains u and v. As we know from Theorem 2, all maximal cliques with uv as the longest edge must lie in F_{uv}. Then we have the following theorem.

Theorem 4 *Any clique with uv as the longest edge must lie in some band B_{uv} within football F_{uv}.*

Proof 4 *Any vertex $x \in F_{uv}$ is $\leq d_{uv}$ from both u and v. So the only way we could violate the theorem is by having two vertices $x, y \in F_{uv}$ that form a clique with u and v, but do not lie in any band. But then $d_{xy} > d_{uv}$ since the height of the band itself is d_{uv}. (Contradiction)*

Thus we will cover every possible maximal clique with uv as the longest edge, if we select *all* such bands B_{uv}. Note that the band B_{uv} may include some extra vertices that are *not* part of the maximal clique as defined. As shown in Fig. 2(b), vertices x and y are included in the band, yet their distance is greater than d_{uv}.

When the vertices in F_{uv} are not all included in either of D_{uv}, $T1_{uv}$ or $T2_{uv}$, we need to use the bands to generate the maximal cliques. The vertices in F_{uv} can be divided into two disjoint groups – those to the north of uv and those to the south (assign boundary nodes to any one). For each vertex $x \in F_{uv}$, depending on whether it is to the north (or south) of uv, we position a band such that its north (or south) boundary lies on x. If the distance of x from uv is less than $1 - \frac{\sqrt{3}}{2}$, we simply position a band at its southernmost (or northernmost) position. There are at most Δ such bands, since u or v has no more than Δ neighbors. Fig. 2(c) shows two such bands $B1_{uv}$ and $B2_{uv}$. In this case we have the following theorem:

Theorem 5 *Choose a set of bands $B^i_{uv}, i = 1, \ldots, \Delta$ corresponding to each vertex in F_{uv}, as described above. Then, every maximal clique with uv as the longest edge, is contained in some B^i_{uv} in this set.*

Proof 5 *Consider any maximal clique $q \in F_{uv}$. Amongst all its vertices, let x be the vertex farthest from the line uv. Then, some band B^j_{uv} is positioned with x on one of its boundaries, and also contains u and v. Since x is the farthest vertex in the clique, all other vertices must lie on the same side of x as uv. But they are all at a distance $\leq d_{uv}$ from x. So all the other members must lie within the band, whose width is d_{uv}. Hence $q \subseteq B^j_{uv}$.*

3.3 Basic Algorithm

findAllMaximalCliquesInUDG:

```
1.  order edges in decreasing order of length;
2.  for each edge uv
3.      if d_uv ≤ 1/√3
4.          output maximal clique F_uv;
5.      else
6.          output three maximal cliques:  D_uv, T1_uv, and T2_uv;
```

```
7.          if F_uv = D_uv or F_uv = T1_uv or F_uv = T2_uv
8.              we are done;
9.          else
10.             for each vertex x ∈ F_uv
11.                 Position band B_uv with boundary on x;
12.                 Vertices in band form clique;
```

3.4 Complexity

We assume that the description of V also contains the geometric locations of the vertices. Let m be the number of edges, and Δ be the maximum degree of the graph. Then, step 2 is evaluated m times. There may be at most Δ vertices in F_{uv}, so step 10 is evaluated Δ times. Finally, in order to generate a clique, we need to look at every neighbor of u and v, and check if they lie within the prescribed region. This may be achieved in at most 2Δ operations. Thus the overall algorithm is $O(m.\Delta.2\Delta) = O(m\Delta^2)$.

For each link, the algorithm generates up to Δ cliques. Thus, the number of cliques generated is $O(m\Delta)$.

3.5 Modified Algorithm

Each clique found in Sec. 3.3 is maximal with uv as its longest edge. However, recall that our motivation is in finding maximal cliques in the entire graph; so any generated clique that is a subset of some other clique is redundant. Suppose that instead of taking the disk D_{uv} with diameter d_{uv}, we consider a potentially larger disk of diameter 1, lying on uv (both disks have the same center). Let D^1_{uv} be the set of vertices in this larger disk, as shown in Fig. 3(a). D^1_{uv} may include edges longer than uv, but these edges are ≤ 1. So D^1_{uv} is a maximal clique in G, and $D_{uv} \subseteq D^1_{uv}$. Thus we in fact do *better* by looking at D^1_{uv} instead of D_{uv}.

Similarly, vertices in the upper curved triangle $T1^1_{uv}$ as shown in Fig. 3(b) (as also the lower curved triangle $T2^1_{uv}$), with sides 1, also forms a maximal clique. $T1^1_{uv}$ shares the same uppermost point with $T1_{uv}$, and so $T1_{uv} \subseteq T1^1_{uv}$. We then have the following theorem.

Theorem 6 *If $d_{uv} \leq \sqrt{3} - 1$, every band B_{uv} is contained in either $F_{uv} \cap T1^1_{uv}$ or $F_{uv} \cap T2^1_{uv}$.*

Proof 6 *We note that $T1^1_{uv}$ and $T2^1_{uv}$ overlap, as shown in Fig. 3(c). For a small enough d_{uv}, the overlap is such that any band B_{uv} is wholly contained in either $T1^1_{uv}$ or $T2^1_{uv}$. This happens when*

$$1 - d_{uv} \geq d_{uv}\sqrt{3} - 1 \;\Rightarrow\; 2 \geq d_{uv}(\sqrt{3}+1) \;\Rightarrow\; d_{uv} \leq \frac{2}{\sqrt{3}+1} = \sqrt{3} - 1 \tag{3}$$

Also, $B_{uv} \subseteq F_{uv}$. Hence for $d_{uv} \leq \sqrt{3} - 1$, every possible B_{uv} is contained in either $F_{uv} \cap T1^1_{uv}$ or $F_{uv} \cap T2^1_{uv}$.

But $T1^1_{uv}$ and $T2^1_{uv}$ are both maximal cliques. Hence it is enough to consider only $T1^1_{uv}$ and $T2^1_{uv}$ instead of looking at all the bands. We use this knowledge to modify the algorithm.

findAllMaximalCliquesInUDGModified:

```
1.   order edges in decreasing order of length;
2.   for each edge uv
3.       if d_uv ≤ 1/√3
4.           output maximal clique F_uv;
5.       else
6.           output three maximal cliques:  F_uv ∩ D^1_uv, F_uv ∩ T1^1_uv, and F_uv ∩ T2^1_uv;
7.           if d_uv ≤ √3 − 1
8.               we are done;
9.           else if no vertex in F_uv \ D^1_uv or F_uv \ T1^1_uv or F_uv \ T2^1_uv
10.              we are done;
11.          else
12.              for each vertex x ∈ F_uv
13.                  Position band B_uv with boundary on x;
14.                  Vertices in band form clique;
```

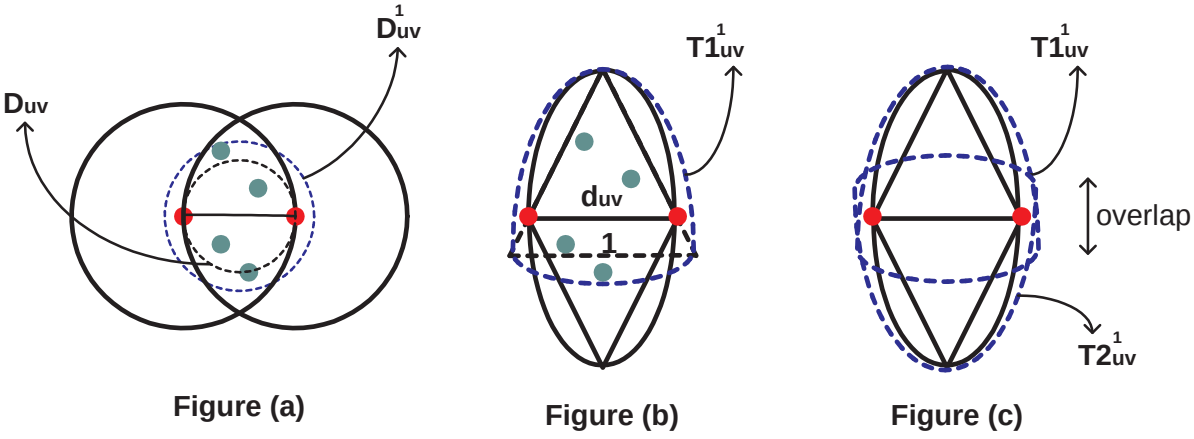


Figure 3: Modified algorithm covering more nodes in maximal clique: (a) Larger band D^1_{uv} (b) Larger curved triangle $T1^1_{uv}$ (c) Overlapping $T1^1_{uv}$ and $T2^1_{uv}$ when d_{uv} is small

The modified algorithm has the same running time $O(m\Delta^2)$ in the worst case as the basic algorithm (Sec. 3.4), and generates the same order of cliques. However, we are more likely to avoid the scanning band in steps 12-14 by checking in steps 7-10 if we have in fact already generated all the maximal cliques we need.

3.6 Analyzing Extra Nodes Covered by Band

In a general graph, it is difficult to assess how many extra nodes the band captures in its cliques. However, it is interesting to analyze the effect in a uniform random graph. In such a case, the number of vertices captured by each covering shape is proportional to its area. We have calculated the areas covered by each of the shapes that we have used. Note that the band has variable area – it is largest when at the middle of the football, and smallest when it is at the extremities.

Shape	Football	Disk	Curved Triangle	Band (max)	Band (min)
Area	$[\frac{2\pi}{3} - \frac{\sqrt{3}}{2}]d^2_{uv} = 1.228d^2_{uv}$	$\frac{\pi}{4}d^2_{uv} = 0.785d^2_{uv}$	$[\frac{\pi}{2} - \frac{\sqrt{3}}{2}]d^2_{uv} = 0.705d^2_{uv}$	$1.008d^2_{uv}$	$0.789d^2_{uv}$

3.7 Handling Changes in the Network

It is important to analyze the evaluation of the algorithm when changes occur in the network. We would like the reaction to these changes to be limited to the locality of the change. We identify five types of changes.

3.7.1 New Vertex

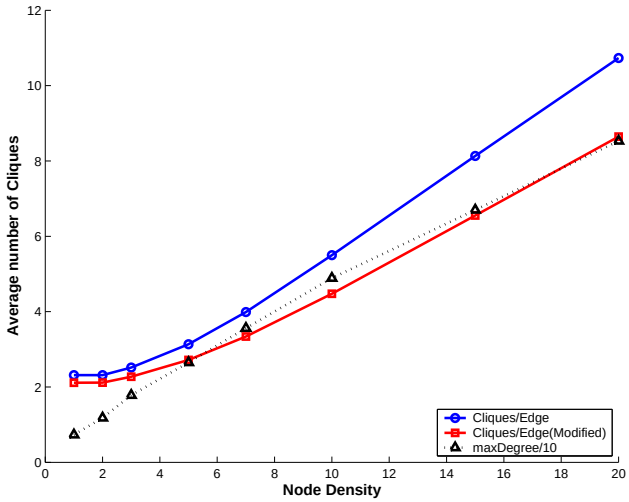
When a new vertex is added to the network, we need to evaluate cliques at all the new edges. This takes $O(\Delta.\Delta^2) = O(\Delta^3)$, since there may be up to Δ edges. The new vertex may add to any of the cliques involving its neighbors; so we need to re-evaluate cliques in the neighborhood of the new vertex. With Δ nodes in the neighborhood, there may be up to Δ^2 edges – hence the overall algorithm is $O(\Delta^2.\Delta^2) = O(\Delta^4)$.

3.7.2 Delete Vertex

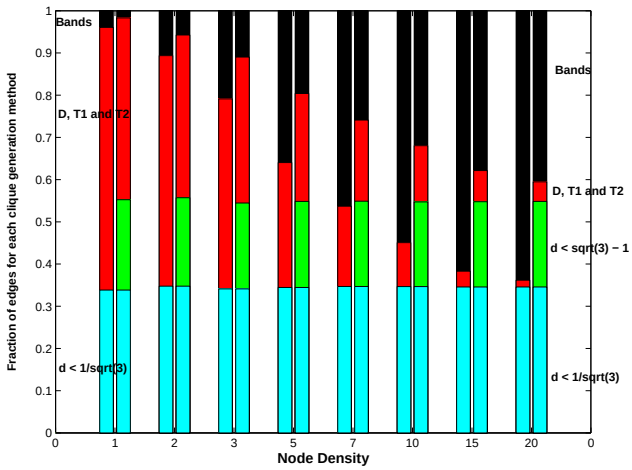
Let the maximal cliques generated be stored in a $q \times n$ clique-node incidence matrix, where q is the number of cliques. From Sec. 3.4, we know that q is $O(m\Delta)$. To delete a vertex i, it suffices to delete the i^{th} column from the matrix. This may be done in $O(m\Delta)$.

3.7.3 New Edge

We need to re-evaluate cliques in the neighborhood of both vertices forming the new edge. There may be up to $2\Delta^2$ edges in the neighborhood, so the overall algorithm is $O(2\Delta^2.\Delta^2) = O(\Delta^4)$.



(a) Number of cliques generated by the approximation algorithms



(b) Comparing the two versions of the algorithm. Left bar = Basic Algorithm. Right bar = Modified Algorithm

Figure 4: Approximating Maximal Cliques in a Random UDG

3.7.4 Delete Edge

Again, we need to re-evaluate cliques at both affected vertices. As above, it takes $O(\Delta^4)$.

3.7.5 Move Vertex

If no new edges are formed or lost, we need not take any steps. However, a change in the network needs to be handled as a Delete Vertex (Sec. 3.7.2), followed by a New Vertex (Sec. 3.7.1). The cumulative algorithm is $O(\Delta^4)$.

For all of the modification operations described above, note that $O(\Delta^2)$ is no greater than $O(m)$, so $O(\Delta^4)$ is at least as good as $O(m\Delta^2)$. Typically $\Delta << m$, so $O(\Delta^4)$ presents a substantial gain over $O(m\Delta^2)$.

3.8 Distributed Algorithm

The maximal clique algorithms presented in Sec. 3.3, 3.5, and 3.7 rely only on *localized* information. For instance, in an ad-hoc network, we would only require each node to know about location and topology in its 2-hop neighborhood. In a practical scenario, the algorithms may thus be implemented in a distributed fashion – with each node computing all maximal cliques around it. This would take $O(\Delta^2)$ time.

4 Simulation Results

We evaluate the performance of our algorithms in a UDG formed by nodes distributed randomly in the plane. We consider a field of dimension 10×10 and uniformly distribute between 100 to 2000 nodes on it, yielding a node density ranging from 1 to 20 per unit square. We then execute the algorithms (both basic and modified) on the generated topologies, the results of which are presented in Fig. 4. Note that each data point on the plot is an average over 10 separate randomly generated topologies, having the same node density.

Fig. 4(a) shows the number of cliques per edge, as the node density increases. We also plot the value of the maximum degree Δ on the same graph, using the dotted line (note that this value is scaled down by a factor of 10, to improve visibility). In a random UDG, Δ increases linearly with node density. Recall that the number of cliques generated by our approximation algorithm is $O(m\Delta)$, thus the number of cliques per edge is $O(\Delta)$. As expected, cliques/edge shows the same shape as Δ. However, the optimizations achieved by using the disks and the curved triangles ensure that the actual number of cliques is significantly less. In fact, the actual cliques/edge generated by the basic algorithm is only around $\Delta/8$. The modified algorithm presented in Sec. 3.5 provides a further reduction in the number of cliques, as shown in the figure.

Fig. 4(b) compares the detailed workings of the two flavors of the algorithm. Using a bar graph, we look at the fraction of edges that are evaluated using each block in the logic of the algorithms, as given in sections 3.3 and 3.5. We distinguish

these as '$d < 1/\sqrt{3}$', '$d < \sqrt{3} - 1$', '$D, T1, T2$', and '$Bands$'. For each value of node density, the left bar denotes the basic algorithm and the right bar denotes the modified algorithm.

Consider the left bar for each node density, documenting the functioning of the basic algorithm. In all the simulations, $1/3$ of the edges (lowest bar, colored cyan) directly yield a single maximal clique F_{uv}, when $d_{uv} < 1/\sqrt{3}$. This is to be expected, since the probability of this is given by $\left(1/\sqrt{3}\right)^2 = 1/3$. Several more edges are handled by one of the special cliques D_{uv}, $T1_{uv}$ or $T2_{uv}$, as indicated by the red bar in the middle. The fraction of edges where we have to resort to the Bands, is shown by the black bar at the very top. As seen from the figure, very few edges require this method when the node density is small, but the fraction of edges increases with node density. At the maximum node density of 20, nearly 65% of the edges require the computation of the Bands.

The modified version of the algorithm introduces another case, whereby the sets $T1_{uv}$ and $T2_{uv}$ suffice when $d_{uv} < \sqrt{3} - 1$. This is denoted by the second bar from the bottom (colored green) for the modified algorithm. This causes a reduction in the fraction of edges requiring the evaluation of Bands (down to about 40% when the node density is 20), and therefore a reduction in the total number of cliques.

5 Conclusions

Wireless and ad-hoc networks are often modelled as unit disk graphs, and clique structures in them are used in several applications. We consider the problem of generating all maximal cliques in a UDG. General algorithms to find cliques in a graph are exponential, so we rely on a polynomial approximation.

We consider each edge, and find all maximal cliques with this as the longest edge. Our algorithm works by making certain key observations about the geometric structure of these graphs. We first limit the possible clique-forming vertices into an area shaped like a football. Then we use two other shapes – the disk and the curved triangle – which we prove to generate cliques. If all cliques have not yet been found, we use a band shape to scan the football to generate all cliques. Our algorithm works in $O(m\Delta^2)$ time and generates $O(m\Delta)$ cliques.

References

[1] A. Graf, M. Stumpf, and G. Weisenfels, "On Coloring Unit Disk Graphs," *Algorithmica*, vol. 20 (1998), pp. 277-293.

[2] R. Gupta, J. Musacchio, and J. Walrand, "Sufficient Rate Constraints for QoS Flows in Ad-Hoc Networks," *UCB/ERL Technical Menorandum M04/42*, Fall 2004.

[3] A. Puri, "Optimizing Traffic Flow in Fixed Wireless Networks," *Proc. WCNC 2002*.

[4] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of Interference on Multi-hop Wireless Network Performance," *ACM Mobicom 2003*, San Diego, CA, USA, September 2003.

[5] Y. Xue, B. Li, and K. Nahrstedt, "Price-based Resource Allocation in Wireless Ad-Hoc Networks," in *Proc. IWQoS 2003*, Monterey, California, June 2003.

[6] F. Harary, and I. C. Ross, "A Procedure for Clique Detection Using the Group Matrix," *Sociometry*, vol. 20, pp. 205-215, 1957.

[7] E. Bierstone, "Cliques and Generalized Cliques in a Finite Linear Graph," Unpublished Report, 1960s.

[8] J. G. Augustson, and J. Minker, "An Analysis of Some Graph Theoretical Cluster Techniques," *Journal of the ACM (JACM)*, vol. 17, no. 4, pp. 571-588, October 1970.

[9] C. Bron and J. Kerbosch, "Finding All Cliques in an Undirected Graph," *Communications of the ACM*, vol. 16, pp. 575–577, 1973.

[10] S. Tsukiyama, M. Ide, H. Ariyoshi, and I. Shirakawa, "A New Algorithm for Generating all the Maximal Independent Sets," *SIAM Journal of Computing*, vol. 6, pp. 505–517, 1977.

[11] G. Yu, O. Goldschmidt, and H. Chen, "Clique, Independent Set, and Vertex Cover in Geometric Graphs," *unpublished report*, 1992.

[12] R. Gupta and J. Walrand, "Approximating Maximal Cliques in Ad-Hoc Networks," *Proc. PIMRC 2004*, Barcelona, Spain, September 2004.