Employing $(1-\varepsilon)$ Dominating Set Partitions as Backbones in Wireless Sensor Networks

Dhia Mahjoub * David W. Matula*

Abstract

For a random geometric graph $G(n,r)$ of minimum degree δ, we introduce an efficient algorithm for selecting $(\delta+1)$ backbones with disjoint node sets that are each independent $(1-\varepsilon)$ dominating sets of G. The backbone node sets are determined by a graph coloring algorithm employing only the topology (not the geometry) of $G(n,r)$, and the backbone links are selected with link lengths in a narrow window between r and $2r$ and further to form a planar graph backbone. For large vertex sets ($n = 1600, 3200$) the resulting backbones are shown to each cover typically over 99% of the vertices of G (i.e. $\varepsilon < 0.01$), with about 30% being fully dominating, which is consistent with the $\frac{1}{4}$ constant approximation factor algorithm proposed recently in [23] for the domatic partition problem in Unit Disk Graphs. We establish experimentally by measures of node degrees, link lengths, and interior triangular face counts that each individual backbone has most of the coverage behavior and routing convenience of the triangular "perfect packing" lattice. We further show for each sample $G(n,r)$ that the relatively few vertices not covered by all $(\delta+1)$ backbones are covered by most of the backbones. Hence backbone rotation in a wireless sensor network would reach all sensors (vertices) sufficiently frequently. Our novel backbone generation algorithm confirms experimentally the existence of these $(\delta+1)$ backbones in a random geometric graph, and provides an efficient topologically based centralized algorithm for determining the backbones. We also point out that our novel backbone construction method is flexible such that any efficient coloring algorithm can be plugged into it. In this paper, we experiment with several coloring algorithms: Smallest Last, Largest First, Lexicographic, Radial Sweep and Random and we compare their respective performance. Our emphasis is, however, on SL since it offers robust properties and interesting expected behavior. We also experiment with several random node distributions: uniform, skewed and normal in both unit square and disk of which we also discuss the results.

*Boby B. Lyle School of Engineering. Southern Methodist University. Dallas, TX 75275-0122, USA. Email: {dmahjoub,matula} @ lyle.smu.edu

1 Introduction

Wireless Sensor Networks are an emerging technology with various military and civilian applications. Sensors are typically thrown in unattended and at random in large numbers on the area to be monitored. Energy efficiency and network lifetime are major design goals in sensor networks and regular clustering is an effective scheme to optimize energy usage in various applications like data gathering. Clustering can be modeled as a connected dominating set (CDS) problem and typical existing approaches strive to optimize one such set in terms of minimum size or maximum node coverage. This approach, however, overtaxes the nodes of the dominating set (known as clusterheads) which leads to their prompt energy depletion. A simplistic approach like in [17] would be to invoke a solution to the CDS problem multiple times to identify disjoint connected dominating sets in a graph. However, this scheme would fail in partitioning a graph into large disjoint CDSs, as the main objective of a CDS algorithm is to minimize the cardinality of the solution. In addition, multiple invocations of the CDS algorithm incur higher latencies to form disjoint connected dominating sets, which may not be a desirable goal. A more perennial scheme is to build several disjoint dominating sets all at once and rotate their activity to gather data and deliver it to the base station so as to balance the load across the network. These multiple disjoint dominating sets can be assimilated to virtual backbones that bring about better scalability and robustness for the benefit of coverage and routing. The problem of virtual backbone rotation can be abstracted as the domatic partition problem, an NP-Complete problem in its decision version and NP-Hard in its optimization one [6].

1.1 Related Work. More generally, the existing work on clustering for coverage and connectivity in Wireless Sensor Networks can be classified into two categories: the first approach attempts to build a single clustered backbone of minimal cardinality where any node should be adjacent to a clusterhead and the clusterheads are regularly separated, with minimal overlap, close to the perfect hexagonal lattice [3][28]. The second

approach attempts to build several disjoint dominating sets that could be duty-cycled or rotated in ensuring coverage and data gathering which in turn balances the energy expenditure over all nodes and extends the network lifetime [11][22]. We believe that the second approach offers better scalability and versatility that can profit various applications in Sensor Networks like coverage, routing, and data aggregation. The problem of finding the maximum number of disjoint dominating sets in an arbitrary graph is known as the domatic partition problem [6]. The majority of the recent work [6][10, 11][20, 21][22, 23, 24][27] focuses on designing centralized and distributed logarithmic or constant factor approximation solutions to the strict domatic partition problem which means the output of the algorithm should be the maximum number of disjoint fully dominating sets, where that maximum is never larger than the minimum degree δ plus one. Some of these works tie in the problem solution to maximizing the clustering or target coverage lifetime in sensor networks [22][27]. For further discussion, we refer the reader to our work in [16]. Recently, the authors of [23] propose the first constant factor approximation algorithm to the domatic partition problem in Unit Disk Graphs. The theoretical result of [23] comes in accordance with our experimental results published around the same time in [16], where we empirically show by using several graph coloring algorithms in Random Geometric Graphs, better results on the domatic number than the $log\ n$ approximation factor of [6][22][27] in arbitrary graphs.

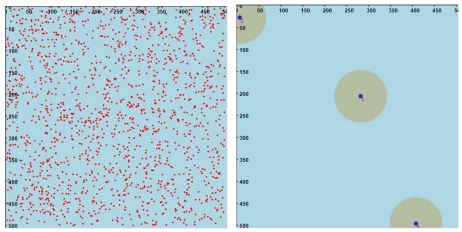
In this paper, we experimentally show the existence with high probability of $(\delta + 1)$ disjoint dominating and almost dominating independent sets and by subsequently building them we propose a novel experimental constant factor approximation solution to the domatic partition problem by using the centralized Smallest Last graph coloring algorithm and the localized Gabriel graph rule applied on Random Geometric Graphs which are a model of choice for Wireless Sensor Networks. Smallest Last is attractive for its efficiency on several classes of graphs [14] but also because it is topology-based rather than geometry based. In fact, in the context of wireless sensor networks, algorithms relying only on connectivity information (graph topology) are preferable when possible to avoid reliance on erroneous localization schemes [5]. To the best of our knowledge, no one attempted to use the graph coloring procedure to systematically investigate the domatic partition problem. Graph coloring provides the useful property of pushing the vertices with the same color far enough apart not to interfere but close enough that we can establish a connected backbone by using 1-hop relays between the backbone's independent vertices. Moreover, by relaxing the total domination constraint, we obtain better practical properties than the strict domatic partition problem solutions. We also experimentally observe that graph coloring produces a reasonably regular packing of the independent vertices of each backbone in dense random geometric graphs.

We also highlight that any efficient coloring algorithm can be utilized as the backbone building component for our method. Therefore, for completeness, we study several coloring algorithms in addition to Smallest Last applied on several random node distributions and compare their respective performance.

1.2 Our Contributions and Outline. Our first contribution is to not only build a nearly optimal number of disjoint independent dominating and almost dominating sets but to also have them regularly packed. Second, we apply the Gabriel graph rule on each virtual backbone to connect its independent vertices via intermediate relays and obtain a planar structure useful for the application of geographic routing algorithms. Third, we consider the problem from the perspective of every sensor (vertex) not being missed very often in any data gathering cycle through the backbones. We evaluate our solution experimentally on large random geometric graph instances and propose several performance measures on the quality of a single backbone and the collection of $(\delta + 1)$ backbones. In particular, we provide measurements on the quality of triangulation of the virtual backbones obtained by the Gabriel graph rule. We also measure the performance of the k-node coverage of every virtual backbone and show the good performance of the backbone rotation concept where very few nodes are successively missed during the data gathering cycles. The rest of the paper is organized as follows. Section 2 defines the graph model and concepts we use in our solution. Section 3 explains our algorithmic approach. Section 4 describes the implementation and the experimental results we obtained, and we finally conclude and propose future work in Section 5.

2 Preliminaries

2.1 Random Geometric Graph Model. A random geometric graph, $G(n, r)$, is formed by placing n vertices uniformly at random in the unit square and connecting two vertices if their Euclidian distance is at most r [16]. Figure 1a shows an example of a random geometric graph (RGG) of 1600 vertices in the unit square with $r = 0.12$. We don't show the edges because the figure would be too crowded. We can define the network density $\mu(r)$ as the expected number of vertices per nominal transmission area: $\mu(r) = n.\pi.r^2$, which also estimates the average vertex degree $\overline{d}$ of any ver-



(a) RGG model (b) Avg. & min degrees

Figure 1: Random Geometric Graph Model.

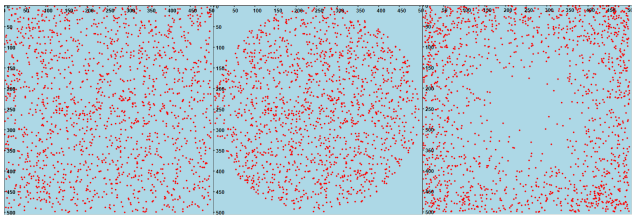
tex whose range is within the unit square. Figure 1b shows sample vertices with their transmission ranges. This gives an estimate of how low the average min degree could be, which is $\frac{n.\pi.r^2}{4}$ in the case the vertex is deployed in the corner of the unit square, i.e. it would have an expected degree equal to about $\frac{1}{4}$ of the average degree of the graph. Similarly, vertices near the boundary but away from the corners should have degrees of about $\frac{n.\pi.r^2}{2}$, or half the average degree. For a random geometric graph $G(1600, 0.12)$, we obtain $\mu(r) = 72.4$, with $\frac{\mu(r)}{4} = 18.1$. Our sample instance in Figure 1a has minimum degree 18 and average degree $\overline{d} = 64.55$. The comparison of $\overline{d}$ with $\mu(r)$ is reasonable since about 40% of the vertices for $r = 0.12$ will have their transmission range touch the border.

2.2 Random Node Distributions. In this work, we consider various random node distributions other than the uniform case. We experiment with the uniform distribution, a skewed distribution and the normal distribution in the unit disk as well as in the unit square. Our results apply to asymptotic average case analysis on graph instances drawn from multiple distributions that may arise in practical sensor network applications.

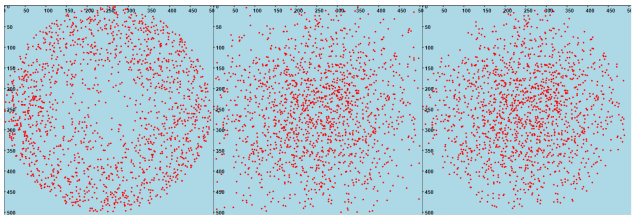
Uniform distribution: For each node, the x and y coordinates are generated uniformly at random.

Skewed distribution: For each node, the x and y coordinates are generated uniformly at random but the node is placed in the deployment area (square or disk) based on its proximity to the center c. A node u that falls on the outer boundary, i.e. $d(u, c) = \sqrt{2}$ in the case of the unit square (in the 4 corners), and $d(u, c) = 1$ in the case of the unit disk (on the circumference), is effectively placed with probability 1. The probability of placing nodes then decreases linearly towards the center where it is 0. This gives denser outer regions and sparser inner regions in the deployment area.

Normal distribution: For each node, the x and y coordinates are generated following a normal distribution $N(\mu, \sigma)$ with μ=0.5, so that nodes are concentrated in the center of the deployment area and σ can vary in [0,1] to represent the spread-out of nodes around the center. In our experiments, $\sigma = 0.24$. The skewed and normal distributions may model sensors deployed respectively on a hill or in a valley, where nodes may be unevenly scattered. We prefer the unit disk model since the disk reduces the relative size of the boundary and provides for the expected degree of vertices near the boundary to be asymptotically close to half the average degree rather than as low as one quarter as pertains to the artifact of vertices near the corners in the unit square boundary model. Figures 2a to 2f show sample graph vertex sets generated according to the different random distributions we studied.



(a) Uniform square (b) Uniform disk (c) Skewed square



(d) Skewed disk (e) Normal square (f) Normal disk

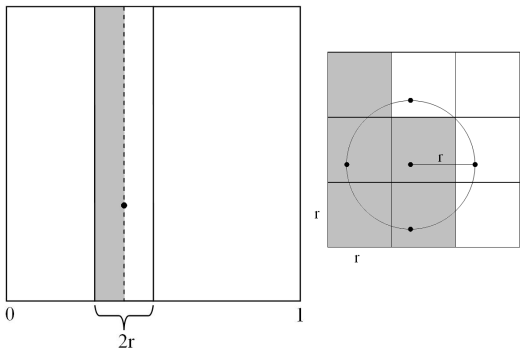
Figure 2: Random Node Distributions.

2.3 Worst Case Node Distributions. Another geometric model of choice for wireless networks is the Unit Disk Graph (UDG). The UDG model represents the situation where all nodes are embedded in the plane and have an identical transmission range (normalized to 1). Whether a communication between two nodes is possible (which translates into an edge in between) depends only on their Euclidian distance. Given a set of n unit disks in the plane which are connected by pairwise overlap, it is evident that the set falls in a disk of diameter n, so that this is equivalent to n points in the unit disk with $r = \frac{1}{n}$. The UDG does not make any assumptions on the random distribution of nodes in the plane. Therefore, results in the literature on the hardness or approximability of graph combinatorial problems in UDGs con-

sider the worst case analysis, which gives the strongest performance guarantees [9].

2.4 Edge Generation Procedures. Once the nodes' x, y coordinates have been generated according to some random distribution, we need to build the adjacency information of the graph. A trivial approach is that every node u checks every other node v of the remaining $(n-1)$ nodes and creates an edge (u, v) if $d(u, v) \leq r$, which runs in $O(n^2)$. A more efficient method termed the *sweep line* or *lexicographic method* first orders the vertices $v_1, v_2, ...$ by their x coordinate and then determines the neighbors of each vertex v_i in this order by moving the window of vertices checked $v_j, v_{j+1}, ..., v_i, v_{i+1}, ..., v_k$ to those with x coordinates having distance from v_i at most r. The expected number of vertices in the resulting vertical strip illustrated in Figure 3a is then $2nr$, so the generation runtime is $O(n^2 r)$. For the constant density model where $nr^2 = c$ as $n \to \infty$, the adjacency list structure for the RGG is obtained in $O(n^{\frac{3}{2}})$ by the *lexicographic sweep line* method. Another method that we term the *radial sweep* algorithm orders the vertices by their distance from the center point $(\frac{1}{2}, \frac{1}{2})$. For a vertex at distance ρ, the vertex is checked against all vertices in the annulus of width $2r$ having area essentially $4\pi r\rho$ (containing essentially $16r\rho n$) vertices for $r << \rho$. For the constant density model, the RGG adjacency information is again obtained in $O(n^{\frac{3}{2}})$. A still more efficient method is the *cell method* where the unit square is partitioned into a grid of $\frac{1}{r^2}$ cells of size r^2 and each vertex is put in the appropriate cell as in bucket sort. The neighbors of each vertex are then determined by checking the distance of the vertex from all vertices of the nine cells including and bounding the cell containing the vertex. The expected number of pairwise distances checked is then $O(n^2 r^2)$ generating the graph in $O(n)$. Furthermore, the *cell method* is optimal in the sense of having expected behavior per vertex of checking only a constant times the average degree of each vertex, where that constant is asymptotically $\frac{9}{\pi}$, as illustrated by the coverage ranges shown in Figure 3b. As illustrated in Figures 3a and 3b, the totality of edges may be found by checking vertex v_i only against others in the shaded vertical strip for the *lexicographic sweep line method* or on the shaded pentonimo for the *cell method* as we methodically traverse the square.

2.5 Backbone Partitions of an RGG. Our goal is to partition a significant portion of the vertices into disjoint sets where each backbone vertex set has vertices that are pairwise at least distance r apart to avoid interference (an *"independent set"*) and sufficiently close



(a) Lexicographic sweep line method (b) Cell method

Figure 3: Edge construction methods.

to cover all or almost all of the unit square or disk, i.e. a *"dominating"* set of all the other vertices. Specifically:

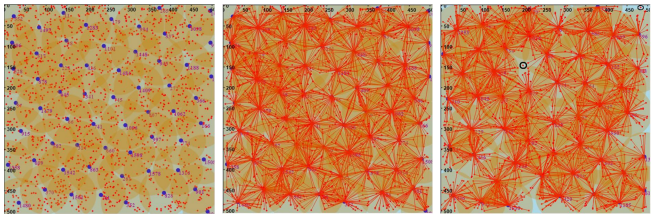
Independent set: In a graph $G = (V, E)$, an independent set S of G is a subset of vertices such that no two vertices $\forall u, v \in S$ are adjacent in G. S is a maximal independent set (MIS) if no vertex can be added to it without violating its independence, and equivalently if any vertex not in S has a neighbor in S, i.e., if S forms a dominating set [16].

Dominating set: In a graph $G = (V, E)$, a dominating set S of G is a subset of vertices such that every vertex of G is either in S or has at least one neighbor in S [4][16]. We define a $(1-\epsilon)$ dominating set S as having at most $\epsilon|V|$ vertices that are not adjacent to members of S, i.e. the $(1-\epsilon)$ dominating set S covers nearly all vertices of G as $(\epsilon \to 0)$. As an absolute measure of domination, we can talk about $(n-k)$ domination where k is the number of vertices being missed, e.g. in Figure 4c the independent set is $(n-2)$ dominating, as it is missing 2 vertices (circled in black). However, in the case k is large but still relatively small compared to n, we can use the relative measure of $(1-\epsilon)$ domination. Figure 4b shows a dominating set of size 61 nodes and Figure 4c shows an almost dominating set of size 53 that misses only 2 vertices. The coverage is still 99.875% and $\epsilon = \frac{2}{1600}$.

The quality of a backbone vertex partition can be measured by the number of dominating sets obtained, the number of $(1-\epsilon)$ dominating sets for various ϵ such as 99.9%, 99%, and the *"yield"*. Specifically:

Yield: For a backbone vertex set partition of a random geometric graph, the yield of $(1-\epsilon)$ domination is the portion of the vertices in the union of the backbone vertex sets each being at least $(1-\epsilon)$ dominating sets of the graph.

Backbone: In the context of wireless networks, a backbone is a subset of nodes that are selected to play a



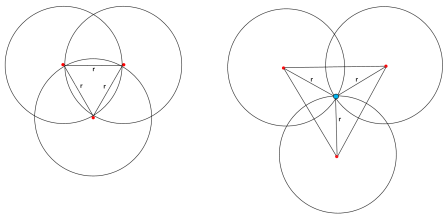
(a) Independent set (b) Dominating set (c) 1-ϵ dominating set

Figure 4: Backbone Model.

special role such as data gathering, routing or data aggregation. Only links within the backbone and direct links from other nodes to the backbone are operating. In the literature, the backbone nodes are termed coordinators, or clusterheads [12], and the backbone can clearly be assimilated to a dominating set. Our backbone model employs vertex sets that are independent and dominating, or $(1-\epsilon)$ dominating for small ϵ.

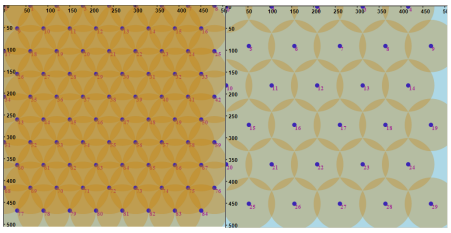
Ideal Backbone: The maximum and minimum number of vertices in a backbone set are conveniently illustrated by considering the triangular lattice resulting from closest packing as traditionally considered in cellular systems. By having the vertices at distance just over r as illustrated in Figure 5a we obtain maximum overlap where all points other than the backbone set are covered at least by three backbone vertices providing redundancy to aid reliability. Alternatively, by having the vertices at $\sqrt{3}r$ we minimize the redundant coverage while still covering the whole region. These triangular lattices (ignoring boundary effects) provide the maximum and minimum size of ideal backbone vertex sets that are both independent and dominating over the region. An ideal backbone set partition where all backbones are maximum dominating sets would yield fewer backbones but provide more robust coverage from each backbone set. If all sets were ideally minimum dominating sets, there would be more backbones (almost 3 times as many) but each would be very sensitive to losing coverage from slight perturbations. Figure 6a illustrates the triangular lattice corresponding to a maximum independent and dominating set ($r = 0.12$) and Figure 6b illustrates the triangular lattice providing a minimum dominating set.

Our approach to determining a backbone vertex set partition by graph coloring assures all backbone vertex sets will be independent. By selecting a preferred coloring algorithm that tends to yield roughly equal size independent sets in an RGG, we verify experimentally that it is possible to obtain a $(1-\epsilon)$ dominating set partition incorporating well over half the vertices, where $(1-\epsilon)$ is generally 99% or better. Partitions with 90% yield for uniform random distributions still are shown to provide over 90% coverage by the "weakest" backbones, which indicates rather surprising robustness for a random placement of sensors.



(a) Maximum coverage (b) Minimum coverage

Figure 5: Extremal local triangular lattice coverage.



(a) Max. Domination (b) Min. Domination

Figure 6: Extremal global triangular lattice domination.

Neighborhood: For a vertex v of a graph G, a neighbor of v is a vertex adjacent to v in G. The neighborhood (or open neighborhood) $N(v)$ of v is the set of neighbors of v [4]. The second order neighborhood or $distance-2$ neighborhood of v is denoted as $N_2(v) = \{u : 0 < dist(u,v) \leq 2\}$, where $dist(u,v)$ is the length of the shortest uv-path in G measured by counting the number of edges in the path [21]. Simply put, $N_2(v)$ is the set of all other vertices within 2 hops away from v. It is provable that in UDGs, $N_2(v) \subseteq N_{ED2}(v)$ where N_{ED2} denotes the set of all other vertices within Euclidian distance 2 (or 2r in the case of RGG) [25]. In our backbone model, we are interested in $distance-2$ neighbors of v that are independent and colored with the same color as v.

Gabriel Graph: Historically, Gabriel graphs were introduced to simplify geographical analysis of monitoring stations' data and have been a popular tool [7, 19]. The Gabriel graph (GG) of a set of nodes with positions in the plane is a planar geometric proximity graph which can be computed locally by each node. Given a set of

vertices V, two vertices from V are connected by an edge in the GG if the disk with these vertices as diameter contains no further vertices from V. [12]. We then define each backbone network as the subgraph of a Gabriel graph on the backbone vertex set where any edge between (u, v) in the Gabriel graph must satisfy $dist(u, v) = 2$ in the original RGG to be a backbone link. Thus, each backbone link will always have Euclidian distance at most $2r$. The Gabriel graph is locally definable, and the backbone is a planar graph since it is a subgraph of the Gabriel graph which was proved in [19] to be planar.

3 Our Backbone Selection Algorithm

The main idea of our backbone selection algorithm is to take as input a random geometric graph $G(n, r)$ whose vertex set's x, y coordinates are generated according to some random distribution (typically uniform) and produce as final output a practically near-optimal collection of disjoint fully and almost dominating sets which yield planar and highly connected backbones for the wireless sensor network modeled as a random geometric graph. The high-level view of our algorithm is as follows:

1. Generate a random geometric graph $G(n, r)$ with vertices' x, y coordinates randomly distributed in the unit square or disk (see 2.2).

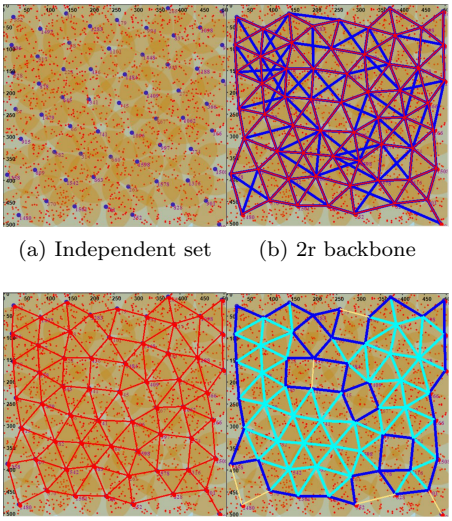
2. Build the adjacency list for every vertex u, where two vertices u, v are adjacent if their Euclidian distance $\leq r$ (see 2.4).

3. Color the graph using an efficient coloring algorithm, typically the Smallest Last algorithm [18]. Smallest Last is topology based, i.e. based solely on vertex adjacency and degree information. The coloring produces k color classes (k independent sets of vertices each with the same color) (see Algorithm 1).

4. For each independent set S from the first $(\delta + 1)$ independent sets, build the $distance - 2r$ graph on $V(S)$ whose links are between vertices $u, v \in S$ that are at Euclidian distance $\leq 2r$ (see Algorithms 2, 3).

5. For each $distance - 2r$ graph built from the first $(\delta + 1)$ independent sets, construct a planar virtual backbone graph by applying the Gabriel graph rule and verifying that u, v are at distance 2 in the original G(n,r) (see Algorithms 2, 3).

6. Classify the backbone edges in three categories: boundary, bridge and interior edges. This will serve in determining the number of triangles in the planar backbone structure, and measure the quality of the triangulation of each backbone.



(a) Independent set (b) 2r backbone

(c) Gabriel backbone (d) Classified links

Figure 7: Independent set, $2r$ backbone (blue/purple links), Gabriel backbone (red links) and classified links.

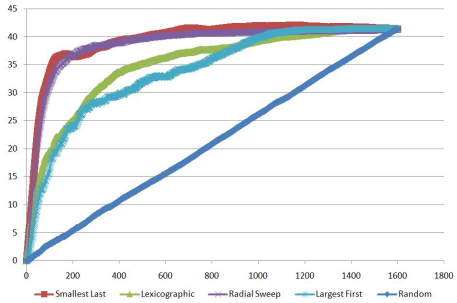
Figure 7 illustrates the backbone vertex set identified by the coloring algorithm that forms a maximal independent set color class one and hence a dominating set of the instance $G(1600, 0.12)$ from Figure 1a, which is then available after steps 1-3. Proceeding to illustrate step 4, Figure 7b illustrates the potential links of the backbone as node pairs whose distance falls in the range $[r, 2r] = [0.12, 0.24]$. To reduce the complexity, in Figure 7c, the blue links are eliminated with the red links remaining satisfying two rules. Each red link identifies a node pair at graph distance two in the graph $G(1600, 0.12)$, i.e. having an intermediate vertex in the overlap region. The red link must also satisfy the Gabriel graph rule that a link between backbone nodes u and v exists iff no other backbone node falls in the circle with diameter $\overline{uv}$ [7][13][19]. Finally, Figure 7d illustrates the result of step 6, where backbone links are classified as to whether they are the boundary of two, one or no triangle in the resulting planar graph.

3.1 Algorithms.

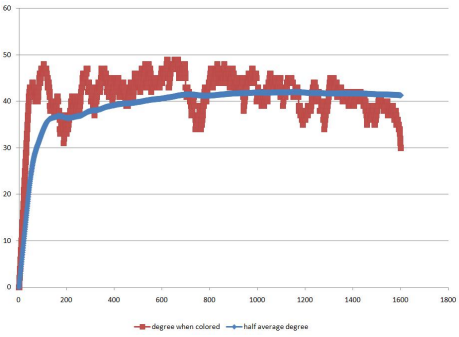
3.1.1 Graph Coloring. The minimum coloring problem is a classical NP-hard problem [14]. In practice, we use heuristics with proven or empirical performance guarantees that determine suboptimal solutions in polynomial time [14]. Most existing coloring heuris-

tics fall in the category of sequential coloring algorithms. A sequential coloring algorithm of graph G is an algorithm that operates in the following two stages: (1) Determine a coloring sequence K of vertices in G. (2) $Greedy-Color(G,K)$. The $Greedy-Color$ procedure, also known as $First-Fit$ or $Grundy\ function$ [8] simply assigns to a given vertex v in the sequence K the smallest color that was not assigned to any neighbor of v previously colored. Our backbone selection method can be built on any efficient graph coloring algorithm. In this study, we mainly focus on the Smallest Last (SL) algorithm and advocate its advantages. In the Experiments section, we compare SL with other coloring algorithms, namely Largest First (LF) [14], Lexicographic [9] and Random. Largest First arranges the vertices in non-increasing order based on degree. Lexicographic arranges vertices in non-decreasing order based on the x coordinate and breaks ties based on the y coordinate. Radial Sweep orders vertices in non-decreasing order based on their distance from the center of the deployment area (square or disk). Random simply assumes a random arrangement of the vertices in the coloring sequence.

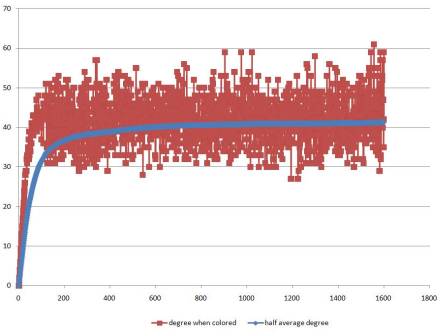
3.1.2 Coloring Sequence Analysis. Our goal is essentially to have each vertex covered by each disjoint backbone vertex set. Essentially, this means we would like the coloring algorithm to assign close to δ distinct colors to the adjacent vertices of each vertex. This process can be aided by ordering the vertices so that when each vertex in the coloring order is assigned a color, there are about the same number of previously colored adjacent vertices where these vertices are sufficiently closely packed to have required close to δ colors in their coloring. Figure 8a illustrates the ratio $\frac{|E|}{|V|}$ in the induced subgraph on the initial sequence of vertices as sequentially colored by the coloring sequences. The random order simply grows $\frac{|E|}{|V|}$ linearly, while the largest first ordering grows different regions responding to their variable densities. The lexicographic line sweep adds vertices and edges reflecting the variation in the semicircle of radius r, adding edges per vertex at an expected mostly constant rate after a smaller startup rate for x values less than r. The radial sweep and smallest last orderings exhibit the most desirable behavior. the radial sweep grows a clique up to radial distance $\rho = \frac{r}{2}$, ramping up the initial average $\frac{|E|}{|V|}$, and then progressively adds vertices of slightly increasing expected degree. The smallest last ordering results in an initial sequence yielding a large complete subgraph generally close or equal to the chromatic number, which is a very desirable feature. The smallest last ordering deletion



(a) Sequential $\frac{|E|}{|V|}$ for all coloring algorithms.



(b) Smallest Last



(c) Radial Sweep

Figure 8: Degree when colored and sequential $\frac{|E|}{|V|}$.

order always removes a min degree vertex. Considering that the convex hull of undeleted vertices has boundary points yielding regions that are less than semicircles, it is apparent that the expected performance of the number of colored adjacent vertices when a vertex is colored by the smallest last order is likely to remain quite tightly bounded. Figure 8b and 8c show the relative stability of this quantity for the smallest last and radial sweep. For these reasons, we focus primarily on the smallest last coloring for backbone vertex set identification, and on input graphs over the unit disk rather than the unit square to avoid the artifact of an initial low minimum degree from a corner vertex of the square.

In general, if A is any graph coloring algorithm then, for a graph G, $\chi_A(G)$ denotes the number of colors that A uses to color G. We define the *performance ratio* of A on G as $\rho_A(G) = \frac{\chi_A(G)}{\chi(G)}$, i.e. the ratio of the number of colors used by A to the number of colors in an optimal coloring of G [15]. Notice that $First - Fit$ combined with any arbitrary vertex ordering computes a 5-approximation coloring in UDGs [2]. By processing the vertices of the graph in a specific order, $First - Fit$ computes 3-approximate solutions in UDGs. For instance, Lexicographic which requires the geometric model of the graph as input and Smallest Last which is purely topological are two 3-approximate vertex orderings [9].

Algorithm 1: Smallest-Last-Coloring

Input: Graph $G(V, E)$
Output: Vertex Ordering K
$K \leftarrow \emptyset$
while $V \setminus K \neq \emptyset$ **do**
  append to K the vertex with smallest degree in the subgraph induced by $V \setminus K$
Greedy-Color(G,$\overline{K}$) //$\overline{K}$ is the inverse order of sequence K

The Smallest Last (SL) method was introduced by Matula et al. [18]. SL is based on the observation that vertices with few neighbors ought to be colored as late as possible [14]. SL computes the vertex ordering of the nodes of a graph G as follows: a node v of minimum degree is removed first and will be colored last. The rest of the ordering is computed recursively on the graph $G \setminus \{v\}$. SL has proven to be efficient on several classes of graphs. In particular, SL is the only known 3-approximation coloring algorithm for UDGs that does not require the graph to be given with its model [9], i.e. it is only based on topology. Furthermore, on planar graphs, SL will also never use more than 6 colors. Notice also that on an RGG in the unit square, first the corner then other boundary nodes will tend to have smaller degree and be removed first. The time complexity of SL is $O(|V| + |E|)$. Figure 9 shows the evolution of the vertex deletion defined by the Smallest Last ordering. From top to bottom the figures show the initial vertex set (without the implicit edges), the first $8(\frac{1}{2}\%)$ to be removed which fall primarily in three corners, the first 200 removed $(\frac{1}{8})$ and the first 400 removed $(\frac{1}{4})$. The third row in red dots shows the $(\frac{n}{2}) = 800$ remaining, followed by the $(\frac{n}{4}) = 400$ remaining, and the $(\frac{n}{8}) = 200$ remaining. The last figure enlarges the region to show the last 36 vertices which form a clique. The last portion of the Smallest Last ordering typically identifies a large clique, which in this case is only one color smaller than the number 37 utilized here to color the whole graph. This confirms that the maximum clique size and chromatic number are each either 36 or 37 for this graph. For our purposes, it is interesting to note that the topology based Smallest Last coloring shrinks the boundary of a contiguous geometric region to a small densely packed region, without explicit use of the geometry of the successive remaining regions.

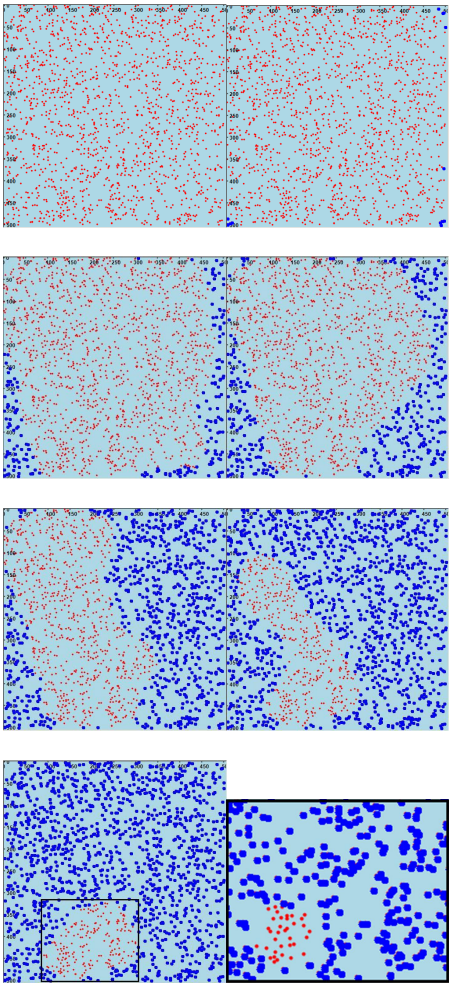


Figure 9: Evolution of the Smallest Last vertex deletion.

3.1.3 Preparing the Backbone. This phase consists in applying two locally-definable procedures: a 2-Hop rule that identifies relay nodes between close-by independent nodes and a Gabriel graph rule which defines backbone virtual links that make it planar.
In Algorithm 2, each node u explores neighboring nodes v at graph distance-2 such that u and v are colored the same, i.e. belong to the same independent set. For

that, since $N_2(v) \subseteq N_{ED2}(v)$ [25], u explores nodes v at *Euclidian distance* $2r$ by expanding its radius from r to $2r$, keeps those nodes v with which it has at least one common neighbor w (relay node) and ignores the rest. This yields a connected backbone of nodes that are at most $2r$ distance apart. Algorithm 2 is locally definable on each node and runs in $O(\Delta^2)$ time in the worst case and $O(\overline{d}^2)$ time in the average case. Each node u requires $O(1)$ storage since the number of distance-2 independent neighbors of u that are colored the same is upper bounded by a constant [1].

In Algorithm 3, each node u checks the set of its immediate neighbors (colored the same) in the connected backbone C obtained from Algorithm 2, and applies the Gabriel graph rule (see definition in 2.5). This produces a connected planar virtual backbone. Algorithm 3 is locally definable and runs on each node in $O(1)$ time since the number of close-by backbone nodes is bounded by a constant.

Algorithm 2: Identify-Relay-Nodes

Input: an independent set of nodes $S(V')$
Output: a connected set of nodes $C(V', E')$:
$\forall u, v \in V'$, $e(u, v) \in E'$ iff
$N(u) \bigcap N(v) \neq \emptyset$

$E' \leftarrow \emptyset$
foreach *node u in S* **do**
  foreach *node v in S*:$d(u, v) \leq 2r$ **do**
    if $\exists$ *node w*:$w \in N(u) \bigcap N(v)$ **then**
      create virtual link $e(u, v)$
      add $e(u, v)$ to E'

Algorithm 3: Build-Gabriel-Links

Input: the connected set of nodes $C(V', E')$ from Algorithm 2
Output: a connected planar virtual backbone $B(V', E'')$
foreach *node u in C* **do**
  foreach *node v in C* **do**
    foreach *node w in C* **do**
      if $w = v$ **then**
        continue
      else if $d(u, v)^2 < [d(u, w)^2 + d(v, w)^2]$ **then**
        eliminate $e(u, v)$ from E'
        break

3.2 Classification of Backbone Links. We define I the set of interior edges, Bo the set of boundary edges, Br the set of bridges and Tr the set of triangular faces in the Gabriel backbone graph. In Figure 7d, a blue edge is an interior edge because it is incident to two triangles, a green edge is a boundary edge as it is incident to one triangle and an orange edge is a bridge since it is not incident to any triangle. We do not need to use a systematic triangle counting algorithm like [26] to find the number of triangles enclosed in the outer face of the backbone. In fact, using the planarity property of the backbone graph, we can make the following observations.

Observation 1: *The sum of the number of boundary edges plus twice the number of interior edges is equal to three times the number of triangles enclosed in the outer face.*

From a counting argument, we can state that: $2|I| + |Bo| = 3|Tr|$ hence $|Tr| = \frac{(2|I|+|Bo|)}{3}$, where an interior edge contributes to 2 triangles and one boundary edge contributes to one triangle.

Observation 2: *The number of boundary edges incident to a vertex v is even. Hence, the path formed by the boundary edges, if connected, constitutes an Euler tour.*

A boundary edge incident to v is in one triangle thereby identifying another edge incident to v that is either a boundary edge or an interior edge. If it is an interior edge it identifies another triangle and another edge incident to v. Continuing this sequence of identifying incident edges that are interior, we must eventually identify another boundary edge as the degree is finite. Thus all boundary edges at v can be paired together by this process, so the number of edges at v must be even. Note that the backbone links illustrated in Figure 7d form such an Euler tour of all nodes incident to a boundary link.

4 Experiments

Implementation Details: The experimentation was done on an Intel Core 2 Duo $E8400$ processor clocked at 3.00 GHz with 3 GB RAM running Windows Vista Enterprise, Service Pack 1. We wrote our own software that simulates the placement of the sensors and implements our proposed solution. The simulation program is implemented in $C\sharp.Net$ (Microsoft Visual Studio 2005) using Windows Forms. The simulator takes the following parameters as input: number of sensors n, and the transmission range r which both determine the average connectivity (density) and places the sensors uniformly at random in the unit square. The distances between all pairs of sensors are then computed and we connect the sensors that are within each other's range.

4.1 Properties and Quality Evaluation of a Single Dominating Backbone. To measure the quality of any single backbone generated by our coloring based

algorithm, it is constructive to compare the resulting backbone with the triangular lattice obtained by a regular closed packing. The lattice shown in Figure 10a has all nodes packed with intra-node distance just over 0.12, and exhibits uniformity except for the discontinuities caused by the outer boundary. Consider the following uniformity measures for the triangular lattice packing.

- Link length uniformity: All lattice links have the same length.
- Degree uniformity: The degree of each interior vertex is 6. There are 2 boundary vertices at the corners of degree 2, and all other boundary vertices have degrees between 3 and 5. The average degree is 5.117.
- Triangular face uniformity: The interior of the outer boundary face is fully triangulated.
- $k-coverage$: Every point in the interior of the outer face other than a lattice node is covered by at least three lattice nodes. Points along the boundary may be only singly or doubly covered. Consider the same uniformity measures for the backbone illustrated in Figure 10b which was generated by our coloring algorithm for a sample random geometric graph $G(1600, 0.12)$ and is a dominating set. Figure 10d indicates the percentage of vertices that are at least covered by k backbone vertices of Figure 10b for $k = 1, 2, 3, 4$.
- The link length distribution falls in a small window between r and $1.6r$, with half of the links of length less than $1.1r$ (see Figure 10c). Notice that the backbone has 151 links. These edge lengths all fall between the bounds $[r, \sqrt{3}r]$ pertaining to the ideal max. and min. triangular lattices of Figures 6a and 6b
- The sample backbone has two boundary vertices of minimum degree 2, with an average degree of 4.870.
- The interior of the outer boundary face includes 88 triangles which is 5 fewer than a full interior triangulation.

In order to measure the typical behavior of a dominating set selected by our coloring algorithm, we applied our algorithm to 10 sample graphs $G(1600, 0.12)$ and utilized the backbone determined by the nodes colored with the first color in each case, where that node set is always a dominating set. For each of these 10 backbones, Table 1 shows the numbers of backbone nodes N and links L, the median link length MLL, and the average degree $\overline{d}$. The link lengths are seen to be of very similar size with half the links on average falling in the interval $[r, 1.1r]$. The average degree is only slightly less than the average 5.117 of the ideal triangular lattice of Figure 10a. The number of interior triangles T in the backbones is seen to average about 85% of the total possible for a full interior triangulation FIT. This data shows that the coloring algorithm for backbone selection employing only the topology of the RGG provides a backbone surprisingly close to that of one obtained by ideal packing. Regarding the k-coverage, we illustrate in Figure 10d that on average our 10 backbone sample provided 2-coverage for about 85% of the other vertices of the sample graphs and at least 3-coverage for over half the vertices.

Table 1: Properties of the first backbone of ten G(1600,0.12) instances.

Instance	N	L	MLL	T	FIT	$\overline{d}$
1	61	144	1.122	76	92	4.721
2	61	147	1.115	83	92	4.820
3	60	138	1.120	72	87	4.600
4	58	139	1.124	76	88	4.793
5	57	130	1.121	64	86	4.561
6	62	149	1.115	83	95	4.806
7	61	140	1.106	73	90	4.590
8	62	147	1.095	82	91	4.742
9	63	151	1.110	84	96	4.794
10	60	139	1.085	72	88	4.633
Average	60.5	142.5	1.111	76.5	90.5	

4.2 Properties and Quality Evaluation of $\delta + 1$ Backbones. Our coloring procedure yields a sequence of backbone node sets $V_1, V_2, ...$ forming $(1 - \epsilon_i)$ dominating sets where i increases modestly up to about 0.01 at $i = \delta$. Figure 11a shows the increase in the average of ε_i for a sample of 100 instances of $G(1600, 0.12)$ and separately for 100 instances of $G(3200, \sqrt{0.5} * 0.12)$, where in both cases the average value of δ is expected to be about 18. The two samples confirm that the growth in ε_i is moderated as n increases given essentially the same average degree, and average minimum degree. These samples extract 20 dominating sets independent of the variation of the average degree $\overline{d}$, since the dominating set behavior is quite stable up to just over a quarter of the average degree, which is more stable over sample instances than the minimum degree δ.

The mean and standard deviation for δ over an initial sample S_i of 100 cases each is shown in Table 2, where we determine that about $\frac{2}{3}$ of the cases ($\overline{\delta}_{int}$)

have minimum degrees between 15 and 20, which is within one standard deviation of the average. For these final samples S_f restricted to the 66 and 68 cases with $15 \leq \delta \leq 20$, Figure 11b illustrates the percentage of individual vertices covered by $\delta, \delta-1, \delta-2...$ backbones. The figure shows that considerably less than 1% of the 1600, respectively 3200 vertices are missed by more than one of the rotating $\delta+1$ backbones.

Table 2: Graph $G(n,r)$ instances used in our experiments.

n, r	$\lvert S_i \rvert$	$\overline{\delta}$	σ	$\overline{\delta}_{int}$	$\lvert S_f \rvert$
$1600, 0.12$	100	17.6	3.23	[15, 20]	66
$3200, \sqrt{0.5} * 0.12$	100	17.18	2.91	[15, 20]	68

Table 3: Domatic partition size and performance ratio in a $G(3200, \sqrt{0.5} * 0.12$ hundred instances sample.

δ	No. Instances	No. Dominating sets			$\frac{Avg.}{\delta+1}$
		Min	Max	Avg.	
15	8	5	8	5.87	0.366
16	17	3	7	5.00	0.294
17	11	3	8	5.00	0.277
18	11	2	8	5.36	0.282
19	13	2	7	5.15	0.257
20	8	4	8	6.25	0.297

Table 3 shows the number out of 100 instances of $G(3200, \sqrt{0.5} * 0.12)$ that have $\delta = 15, 16..., 20$, which constitute the instances where δ is within one standard deviation of the mean. For each of these samples with δ fixed, we tabulate the minimum, maximum, and average number of backbone node sets that are fully dominating. Typically over $\frac{\delta+1}{4}$ backbone node sets are dominating sets of $G(3200, \sqrt{0.5} * 0.12)$, showing our algorithm experimentally confirms the $\frac{1}{4}(\delta+1)$ theoretical lower bound in [23].

For various values of n and r and various distributions in the square and disk (US: Unit Square, UD: Unit Disk, SS: Skewed Square, SD: Skewed Disk, NS: Normal Square, and ND: Normal Disk), Table 4 tabulates the average over 20 instances of the minimum degree, maximum degree and average degree. It also tabulates the initial colored complete subgraph size for the smallest last coloring, which is a good lower bound on the chromatic number for these graphs.

For the test set of graphs of Table 4, Tables 5, 6, 7, 8 and 9 tabulate for the different coloring algorithms we studied the average number of colors employed $\chi_A(G)$, the number of fully dominating sets obtained $\#DS$, the average percentage domination for the first $\delta+1$ colors defined as $\overline{dom_{\delta+1}}$ and the performance ratio $\rho_{DP}(G)$ on the domatic partition size. $\rho_{DP}(G)$ is equal to $\frac{\#DS}{\lfloor\delta\rfloor+1}$ where δ is the average minimum degree from Table 4. Notice that when we indicate a * as superscript to a value, it means the value represents the percentage domination over the first $3(\delta+1)$ backbones since for certain skewed distributions, we obtain graphs that are highly dense in regions (high Δ) but sparse in others (very low δ). The $\delta+1$ performance was not indicative of the overall quality of the obtained backbones, and we observe that the first $3(\delta+1)$ backbones still offer a very good quality domination (over 99%).

5 Conclusions and Future Work

We are in the process of running more comparative tests and plan to investigate more completely the smallest last algorithm in comparison to algorithms utilizing geometric properties. It is intended that we will follow up in the near future, with more tests and their analysis. Overall, the results presented in this paper make use of a centralized coloring algorithm in addition to the Gabriel graph rule which is inherently localized. Distributed algorithms for the connectivity and coverage problem in sensor networks are more desirable given their scalability and robustness. Our current focus is to work on a more rigorous theoretical analysis to prove that our method delivers good results with certain performance guarantees. We are also working on developing a practical distributed variant of our proposed solution, and compare it with similar distributed solutions to the same problem. Notice that distributed SL takes $\Omega(n)$ rounds [8], therefore, we are working on utilizing distributed coloring algorithms that could relax the efficiency of coloring to terminate in a sublinear or constant number of rounds and yet deliver practical good results in terms of $(\delta+1)$ disjoint dominating and almost dominating sets.

Acknowledgment

The authors would like to thank Dr. Saeed Abu-Nimeh for his valuable advice in preparing this document and the anonymous reviewers for helpful comments and corrections to an earlier draft.

References

[1] K. M. Alzoubi, P. Wan, and O. Frieder, "Message-optimal connected dominating sets in mobile ad hoc networks," in *Proc. of MOBIHOC'02*, 2002, pp. 157–164.

[2] I. Caragiannis, A. V. Fishkin, C. Kaklamanis, and E. Papaioannou, "A tight bound for online colouring of disk graphs," *Theoretical Computer Science*, vol. 384, pp. 152–160, 2007.

[3] H. Chan and A. Perrig, "Ace: An emergent algorithm for highly uniform cluster formation," in *Proc. of the First European Workshop on Sensor Networks (EWSN)*, 2004, pp. 154–171.

[4] G. Chartrand and P. Zhang, *Introduction to Graph Theory*. McGraw-Hill Higher Education, 2004.

[5] A. Efrat, D. Forrester, A. Iyer, and S. G. Kobourov, "Force-directed approaches to sensor localization," in *Proc. of the 8th Workshop on Algorithm Engineering and Experiments (ALENEX)*, 2006, pp. 108–118.

[6] U. Feige, M. M. Halldorsson, G. Kortsarz, and A. Srinivasan, "Approximating the domatic number," *J. Of Computing*, vol. 32, no. 1, pp. 172–195, 2003.

[7] K. Gabriel and R. Sokal, "A new statistical approach to geographic variation analysis," *Systematic Zoology*, vol. 18, pp. 259–278, 1969.

[8] C. Gavoille, R. Klasing, A. Kosowski, L. Kuszner, and A. Navarra, "On the complexity of distributed graph coloring with local minimality constraints," *Networks*, vol. 54, no. 1, pp. 12–19, 2009.

[9] A. Gräf, M. Stumpf, and G. Weinβenfels, "On coloring unit disk graphs," *Algorithmica*, vol. 20, pp. 277–293, 1998.

[10] K. Islam, S. G. Akl, and H. Meijer, "Distributed generation of a family of connected dominating sets in wireless sensor networks," in *Proc. of the Fifth IEEE/ACM International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2009, pp. 343–355.

[11] ——, "Maximizing the lifetime of a sensor network through domatic partition," in *Proc. of the 34th IEEE Conference on Local Computer Networks (LCN)*, 2009.

[12] H. Karl and A. Willig, *Protocols and Architectures for Wireless Sensor Networks*. Wiley, 2005.

[13] B. Karp and H. T. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *Proc. of the 6th International Conference on Mobile Computing and Networking (ACM Mobicom)*, 2000, pp. 243–254.

[14] M. Kubale, *Graph Colorings*. American Mathematical Society, 2004.

[15] L. Lovász, M. Saks, and W. T. Trotter, "An on-line graph coloring algorithm with sublinear performance ratio," *Discrete Mathematics*, vol. 75, pp. 319–325, 1989.

[16] D. Mahjoub and D. W. Matula, "Experimental study of independent and dominating sets in wireless sensor networks," in *Proc. Of WASA '09*, ser. LNCS, vol. 5682, 2009, pp. 32–42.

[17] M. Marta, Y. Yang, and M. Cardei, "Energy-efficient composite event detection in wireless sensor networks," in *Proc. Of WASA '09*, ser. LNCS, vol. 5682, 2009, pp. 94–103.

[18] D. W. Matula and L. Beck, "Smallest-last ordering and clustering and graph coloring algorithms," *J. of the ACM*, vol. 30, no. 3, pp. 417–427, 1983, see also D. W. Matula et al., Graph Coloring Algorithms in *Graph Theory and Computing*, Academic Press, 1972, pp. 109-122.

[19] D. W. Matula and R. Sokal, "Properties of gabriel graphs relevant to geographic variation research and the clustering of points in the plane," *Geographical Analysis*, vol. 12, pp. 205–222, July 1980.

[20] R. Misra and C. A. Mandal, "Clusterhead rotation via domatic partition in self-organizing sensor networks," in *Proc. of the Second International Conference on COMmunication System softWAre and MiddlewaRE (COMSWARE 2007)*, 2007.

[21] ——, "Rotation of cds via connected domatic partition in ad hoc sensor networks," *IEEE Trans. Mob. Comput.*, vol. 8, no. 4, pp. 488–499, 2009.

[22] T. Moscibroda and R. Wattenhofer, "Maximizing the lifetime of dominating sets," in *Proc. Of 5th IEEE WMAN'05*, 2005.

[23] S. Pandit, S. V. Pemmaraju, and K. Varadarajan, "Approximation algorithms for domatic partitions of unit disk graphs," in *Proc. of APPROX 2009*, ser. LNCS, vol. 5687, 2009, pp. 312–325.

[24] S. V. Pemmaraju and I. A. Pirwani, "Energy conservation via domatic partitions," in *Proc. Of 7th ACM MOBIHOC' 06*, 2006, pp. 143–154.

[25] T. Ren, K. L. Bryan, and L. Thoma, "On coloring the square of unit disk graph (draft)," 2006.

[26] T. Schank and D. Wagner, "Finding, counting and listing all triangles in large graphs, an experimental study," ser. LNCS, vol. 3503, 2005, pp. 606–609.

[27] M. T. Thai, Y. Li, F. Wang, and D.-Z. Du, "O(log n)-localized algorithms on the coverage problem in heterogeneous sensor networks," in *Proc. of the 26th IEEE International Performance Computing and Communications Conference (IPCCC 2007)*, 2007, pp. 85–92.

[28] H. Zhang and J. C. Hou, "Maintaining sensing coverage and connectivity in large sensor networks," *J. Of Ad Hoc and Sensor Wireless Networks*, vol. 1, no. 1-2, pp. 89–124, 2005.

Table 4: Properties of experimented graph instances.

$Distr., n, r$	δ	Δ	$\bar{d}$	$\omega_{SL}(G)$
US, 800, 0.17	17.6	91.65	62.51	33.35
US, 1600, 0.12	18	95.7	64.97	36.15
US, 3200, 0.084	17.9	98.05	65.47	35.5
UD, 800, 0.17	31.7	114	79.51	40.45
UD, 1600, 0.12	31.35	120.25	82.95	41.45
UD, 3200, 0.084	30.45	118	83.66	42.85
SS, 800, 0.17	10.5	117.2	66.41	51.7
SS, 1600, 0.12	4.7	144.55	75.14	56.35
SS, 3200, 0.084	2.65	164.5	80.74	59.6
SD, 800, 0.17	27.25	116.6	77.02	45.85
SD, 1600, 0.12	19.85	130.5	83.67	49.4
SD, 3200, 0.084	13.1	137.4	87.21	50.4
NS, 800, 0.17	4.2	197.2	107.71	65.7
NS, 1600, 0.12	2.4	214.05	111.05	69.35
NS, 3200, 0.084	1.25	221.2	111.25	72.35
ND, 800, 0.17	15.85	205.4	116.19	67.7
ND, 1600, 0.12	11.7	224.4	120.50	71.55
ND, 3200, 0.084	9.45	231.8	120.63	75.2

Table 5: Performance of Smallest Last.

$Distr., n, r$	$\chi_A(G)$	$\#DS$	$\overline{dom_{\delta+1}}$	$\rho_{DP}(G)$
US, 800, 0.17	36.6	8.4	99.67	0.46
US, 1600, 0.12	38.15	6.95	99.78	0.36
US, 3200, 0.084	38.7	5	99.86	0.27
UD, 800, 0.17	43.95	12.95	99.31	0.40
UD, 1600, 0.12	45.6	10	99.46	0.31
UD, 3200, 0.084	45.85	8.25	99.67	0.26
SS, 800, 0.17	53	4.1	99.7	0.37
SS, 1600, 0.12	85.25	2.4	99.34*	0.48
SS, 3200, 0.084	62.5	1.6	99.73*	0.53
SD, 800, 0.17	48.1	11.55	99.46	0.41
SD, 1600, 0.12	52.1	8.65	99.76	0.43
SD, 3200, 0.084	53.8	5.85	99.90	0.41
NS, 800, 0.17	70.75	1.9	98.73*	0.38
NS, 1600, 0.12	75.35	1.55	99.47*	0.51
NS, 3200, 0.084	78.85	1.25	99.77*	0.62
ND, 800, 0.17	73.55	4.65	99.49	0.29
ND, 1600, 0.12	79.2	3.2	99.77	0.26
ND, 3200, 0.084	81.9	2.2	99.87	0.22

Table 6: Performance of Largest First.

$Distr., n, r$	$\chi_A(G)$	$\#DS$	$\overline{dom_{\delta+1}}$	$\rho_{DP}(G)$
US, 800, 0.17	40.9	6	99.61	0.33
US, 1600, 0.12	42.15	4.7	99.71	0.24
US, 3200, 0.084	42.5	3	99.79	0.16
UD, 800, 0.17	50.05	11.35	99.28	0.35
UD, 1600, 0.12	51	7.45	99.38	0.23
UD, 3200, 0.084	51.9	4.95	99.59	0.15
SS, 800, 0.17	56.3	3.1	99.58	0.28
SS, 1600, 0.12	63.85	1.95	99.22*	0.39
SS, 3200, 0.084	68.65	1.6	99.70*	0.53
SD, 800, 0.17	52.8	7.4	99.06	0.26
SD, 1600, 0.12	58.15	5.15	99.64	0.25
SD, 3200, 0.084	59.4	3.9	99.85	0.27
NS, 800, 0.17	75	2.35	98.76*	0.47
NS, 1600, 0.12	80.8	1.5	99.51*	0.50
NS, 3200, 0.084	87.65	1.3	99.78*	0.65
ND, 800, 0.17	77.45	4.55	99.55	0.28
ND, 1600, 0.12	84.5	3.3	99.77	0.27
ND, 3200, 0.084	91.45	2.8	99.87	0.28

Table 7: Performance of Lexicographic.

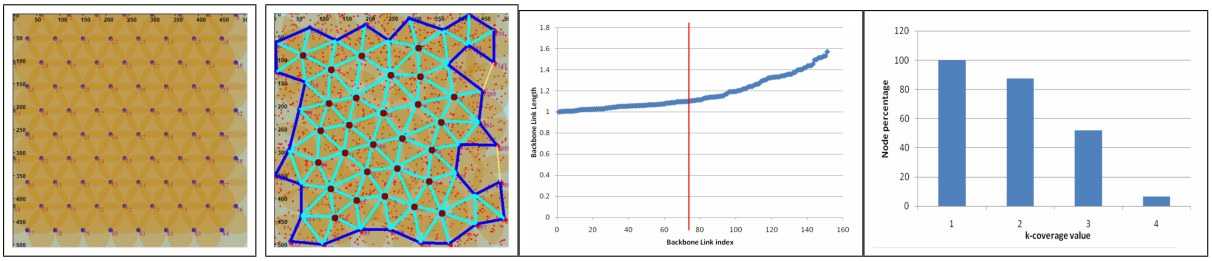
$Distr., n, r$	$\chi_A(G)$	$\#DS$	$\overline{dom_{\delta+1}}$	$\rho_{DP}(G)$
US, 800, 0.17	40.45	10.3	99.72	0.57
US, 1600, 0.12	42.8	9.35	99.79	0.49
US, 3200, 0.084	43.5	7.85	99.87	0.43
UD, 800, 0.17	49.05	14.15	99.00	0.44
UD, 1600, 0.12	51.2	13.15	99.25	0.41
UD, 3200, 0.084	52.45	10.95	99.50	0.35
SS, 800, 0.17	58	6.1	99.83	0.55
SS, 1600, 0.12	65.2	3.15	99.37*	0.63
SS, 3200, 0.084	70.75	2.3	99.76*	0.76
SD, 800, 0.17	53.3	10.6	99.19	0.37
SD, 1600, 0.12	58.35	9.05	99.74	0.45
SD, 3200, 0.084	60.7	6.5	99.90	0.46
NS, 800, 0.17	80.5	2.85	98.91*	0.57
NS, 1600, 0.12	84.75	1.7	99.59*	0.56
NS, 3200, 0.084	88.95	1.5	99.82*	0.75
ND, 800, 0.17	84.2	6.75	99.65	0.42
ND, 1600, 0.12	88.3	4.75	99.83	0.39
ND, 3200, 0.084	92.6	3.5	99.89	0.35

Table 8: Performance of Radial Sweep.

$Distr., n, r$	$\chi_A(G)$	$\#DS$	$\overline{dom_{\delta+1}}$	$\rho_{DP}(G)$
US, $800, 0.17$	39.45	7.3	99.65	0.40
US, $1600, 0.12$	42.2	6.7	99.76	0.35
US, $3200, 0.084$	42.3	4.9	99.83	0.27
UD, $800, 0.17$	47.65	12.05	99.22	0.37
UD, $1600, 0.12$	49.65	9.9	99.34	0.30
UD, $3200, 0.084$	50.65	8.05	99.56	0.25
SS, $800, 0.17$	58.1	6.65	99.85	0.60
SS, $1600, 0.12$	65.2	4.35	99.36*	0.87
SS, $3200, 0.084$	69.95	2.95	99.77*	0.98
SD, $800, 0.17$	53.9	11.85	99.05	0.42
SD, $1600, 0.12$	58.8	9.2	99.67	0.46
SD, $3200, 0.084$	59.95	6.4	99.89	0.45
NS, $800, 0.17$	74.25	1.9	98.69*	0.38
NS, $1600, 0.12$	79.25	1.45	99.47*	0.48
NS, $3200, 0.084$	83.85	1.15	99.77*	0.57
ND, $800, 0.17$	77.1	4.7	99.55	0.29
ND, $1600, 0.12$	83.05	3.2	99.74	0.26
ND, $3200, 0.084$	86.75	2.15	99.85	0.21

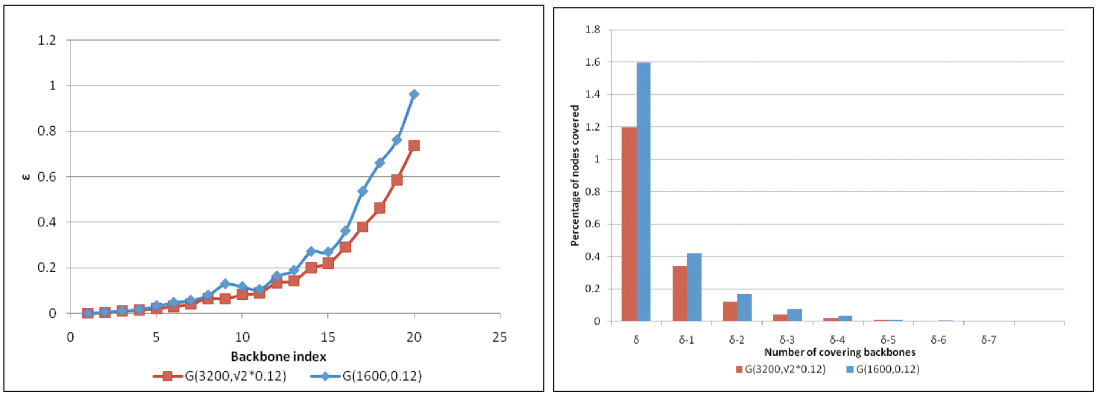
Table 9: Performance of Random.

$Distr., n, r$	$\chi_A(G)$	$\#DS$	$\overline{dom_{\delta+1}}$	$\rho_{DP}(G)$
US, $800, 0.17$	45.2	6.8	99.57	0.37
US, $1600, 0.12$	47.2	5.25	99.65	0.27
US, $3200, 0.084$	48.25	3.5	99.78	0.19
UD, $800, 0.17$	55	9.65	99.05	0.30
UD, $1600, 0.12$	57.2	7.25	99.33	0.22
UD, $3200, 0.084$	57.35	5.7	99.53	0.18
SS, $800, 0.17$	58.75	4.45	99.75	0.40
SS, $1600, 0.12$	66.2	2.35	99.34*	0.47
SS, $3200, 0.084$	73.3	1.6	99.74*	0.53
SD, $800, 0.17$	56	9.4	99.22	0.33
SD, $1600, 0.12$	61.5	6.65	99.72	0.33
SD, $3200, 0.084$	64	4.6	99.87	0.32
NS, $800, 0.17$	87.9	2.2	98.72*	0.44
NS, $1600, 0.12$	96.5	1.35	99.48*	0.45
NS, $3200, 0.084$	102.25	1.1	99.77*	0.55
ND, $800, 0.17$	91.35	4.15	99.44	0.25
ND, $1600, 0.12$	100.15	3.25	99.71	0.27
ND, $3200, 0.084$	105.9	2.45	99.84	0.24



(a) Triangular lattice (b) Triangulated backbone (c) Links length distribution (d) k-coverage

Figure 10: Quality measures for a single dominating backbone.



(a) ϵ as n grows (b) Missed domination as n grows

Figure 11: Performance of the dominating and almost dominating $\delta + 1$ backbones.