Iterative Local Solutions for Connected Dominating Set in Ad Hoc Wireless Networks

Jie Wu†, Fei Dai‡, and Shuhui Yang†
†Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431
‡Department of Electrical and Computer Engineering
North Dakota State University
Fargo, ND 58105

***Abstract*— We propose a general framework of the *iterative local solution* (ILS) for computing a connected dominating set (CDS) in ad hoc wireless networks, which include wireless sensor networks (WSNs) and mobile ad hoc networks (MANETs). This approach uses an iterative application of a selected local solution. Each application of the local solution enhances the result obtained from the previous iteration, but each is based on a different node priority scheme. Then, we integrate this iterative process into the process for handling dynamic network topology and propose two extensions: *cyclic iterative local solution* (CILS) and *seamless iterative local solution* (SILS). CILS offers a natural extension of ILS to the dynamic environment, but suffers from broken CDS and non-adaptiveness. With a novel use of a monotonically increasing sequence number for dynamic node priority, SILS offers an extension with the desirable properties of correctness, progressiveness, locality, and seamlessness. Extensive simulations are conducted to evaluate the effectiveness of the proposed approach in both static and dynamic environments.**

I. Introduction

In ad hoc wireless networks, which includes wireless sensor networks (WSNs) and mobile ad hoc networks (MANETs), various algorithmic solutions can be classified into *global, quasi-global, quasi-local,* and *local* [20] depending on the amount of information used by each node to determine a solution for a specific problem (such as connected dominating set (CDS) as a virtual backbone in MANETS [19] and for coverage in WSNs [9], and network topology control for saving energy and reducing signal interference in MANETs [3]). Dominating sets (DS) have been widely used in the selection process of active node sets in ad hoc wireless networks. A set is dominating if every node in the network is either in the set or a neighbor of a node in the set. When active nodes form a dominating set, all nodes in the network are also said to be reachable. When a DS is connected, where any two nodes in the DS can be connected through intermediate nodes from the DS, it is denoted as a connected dominating set (CDS).

The work was supported in part by NSF grants ANI 0083836, CCR 0329741, CNS 0422762, CNS 0434533, and EIA 0130806. Corresponding author: jie@cse.fau.edu

The local approach uses local information to determine node status and such status does not propagate; i.e., the status of each node does not depend on the status of its neighbors. Therefore, the local approach is the most desirable to support scalable design through localized maintenance in a dynamic environment (also called locality). In the construction of a CDS, the status of a node is either inside or outside the selected CDS; whereas in network topology control, the status of each node is the selected transmission range for the node.

One potential problem of local solutions is low efficiency (i.e., the quality of results). In CDS construction the quality is measured by the size of CDS and in topology control it is measured by the transmission range subject to connectivity. In this paper, we present a general framework of the *iterative local solution* (ILS) that relaxes the non-propagation constraint of local solutions to improve efficiency. Each application of a selected local solution enhances the result obtained from the previous iteration, but based on a different node priority scheme. However, ILS still keeps locality; that is, ILS can quickly provide a solution after a network topology change.

Figure 1 shows the difference between global, local, and iterative local solutions where time is slotted into rounds each of which is a square block. Each round is measured as one or more “Hello” message exchanges in ad hoc wireless networks. To simplify the discussion, local solutions take one round to generate a solution. Both gray and black blocks correspond to correct results generated at respective rounds, with the darker the color of a block, the higher the efficiency of the result generated at the corresponding round. ILS can quickly generate a result, albeit inefficient, and then improves it over iterations before the next network topology change (represented by a vertical line in the figure). In global solutions, an efficient solution can be generated after several rounds (say r). However, if the network topology changes frequently, no results can be generated in global solutions as in Figure 1

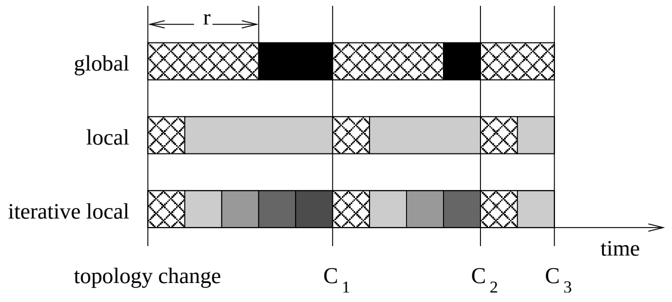


Fig. 1. Comparison among global, local, and iterative local solutions.

where the distance between two changes (c_2 and c_3) is less than r. Note that in ILS nodes exchange new node priority and node status between rounds while in global solutions nodes exchange link state information.

In this paper, we focus on using ILS to calculate a CDS, with the objective of reducing the CDS size over a number of iterations. Here ILS is first discussed in a static environment, followed by its extensions in a dynamic environment. This framework is illustrated using Dai and Wu's Rule K [6], an extension of Wu and Li's marking process [19], as a local solution. Each node determines its status: *marked* (inside CDS) or *unmarked* (outside CDS), based on local topology information and node priority in the neighborhood. Basically, a node can be unmarked if its neighborhood can be covered (dominated) by a set of nodes with higher priorities and these nodes are connected by themselves.

When applying ILS to a static environment, each node collects topology information within h hops (for a small constant h), collecting h-hop information in what is called one round (iteration), and then determines its node status, marked or unmarked, through an iterative process with a constant number (k) of iterations. During each iteration, nodes are assigned different priorities so that more nodes can be unmarked as the process iterates. When applying ILS to a dynamic environment, the challenges lie in seamlessly blending topology changes into the scheme so that the following properties are maintained:

- **Correctness**: The CDS should be maintained at the end of each iteration (round) unless a new topology change occurs during the iteration.
- **Progressiveness**: The CDS size should be monotonically decreasing between iterations when there is no topology change.
- **Locality**: A topology change only affects the status of nodes in the local neighborhood, where the hop count in such a neighborhood depends on h.
- **Seamlessness**: The effects of the iterative process and topology change are integrated in a seamless way.

We propose two extensions of ILS: *cyclic iterative local solution* (CILS) and *seamless iterative local solution* (SILS). CILS offers a natural extension of ILS to the dynamic environment where ILS is cyclically applied for every k iterations. However, none of the above properties can be maintained. With a novel use of a monotonically increasing sequence number for dynamic node priority, SILS offers an extension with the desirable properties of correctness, progressiveness, locality, and seamlessness. Due to the space limit, all proofs of the theorems in this paper are omitted.

II. Related Work

Our objective is to find a CDS that covers a unit disk graph representing a MANET based on local information. The problem of finding a minimum CDS (MCDS) is NP-complete for both general graphs [4] and unit disk graphs [12]. Heuristic algorithms to construct a CDS fall into four groups: global [7], quasi-global [17], quasi-local [20], and local [14], [16], [18], [19]. Many local solutions rely on node priorities to avoid simultaneous withdrawals in mutual coverage cases. One drawback of these priority-based schemes is that they may select a large CDS based on a bad priority assignment. Attempts have been made to mitigate this problem. For example, Stojmenovic [13], [14] proposed to reduce the CDS size via adaptive interpretation of priority values. In these schemes, the priority assignment is fixed; therefore, they cannot effectively eliminate redundant dominating nodes. In [1], a mechanism is applied to dynamically maintain the CDS property in a dynamic environment, not dynamically reduce the CDS size over time.

Several iterative approaches have been proposed to find a small DS [8], [10] or CDS [11] in MANETs. In [8], Gao et al gave a basic algorithm to find a DS with an expected approximation ratio of $O(\sqrt{n})$, where each node designates a node with the highest priority in its neighborhood as a dominator. To obtain an expected $O(1)$ approximation ratio, the basic algorithm is repeated $\log(\log n)$ times using exponentially growing transmission ranges. In another iterative DS algorithm proposed by Kuhn and Wattenhofer [10], each node v becomes a dominator with a probability p_v. If there are still uncovered nodes (i.e., nodes without neighboring dominators) after this process, these uncovered nodes also become dominators. The probability p_v is computed via a distributed linear programming algorithm that takes k^2 iterations with an adjustable parameter k. The iterative algorithm has an expected approximation ratio of $O(k\Delta^{2/k}\log\Delta)$, where Δ is the maximal node degree.

Liu, Pan, and Cao [11] proposed an iterative extension of Wu and Li's marking process and Rules 1 and 2 [19] for the local construction of a CDS. In the marking process, a node becomes a dominator (*marked*) if it has two neighbors that are not directly connected. According to Rule 1, a marked node can change back to a non-dominator (unmarked), if all its neighbors are also neighbors of another marked node with a higher priority (called a *coverage node*). In Rule 2, a marked node can be unmarked if its neighbor set is covered jointly by two connected coverage nodes. The iterative extension takes six rounds. The marking process is applied in round 1. Rule 1 is applied in round 2 with one priority (lower node ID has a higher priority) and round 3 with another one (higher node ID has a higher priority). Finally, Rule 2 is applied in rounds 4, 5, and 6 with different priority functions. This approach produces a smaller CDS than the original marking process and Rules 1 and 2.

None of the above approaches address the CDS maintenance issue in dynamic networks, where topology changes, such as link switched-on/off and node switched-on/off, occur during the iterative process. This paper proposes an iterative scheme that integrates the CDS maintenance mechanism into the iterative CDS reduction process, and maintains a CDS at each round of iteration.

III. Iterative Local Solution (ILS) in a Static Environment

This section starts with a general model for iterative local solution (ILS), which extends a scheme proposed by Liu, Pan, and Cao [11]. Dai and Wu's Rule K [6] is used to illustrate the model. The section ends with a discussion of various ways of generating dynamic node priority based on node IDs.

A. General model

Algorithm 1 k-round Iterative Local Solution (at each node v)

```
1: Each node collects local topology information and applies a local solution to determine its status (marked or unmarked).
2: The process completes if the number of iterations reaches $k$; otherwise, each node selects a new priority and exchanges status (and priority if needed) with neighbors.
3: Apply the local solution again based on new node status and node priority. Go to step 2 for the next iteration.
```

Algorithm 1 shows a k-round ILS, where local topology information can be defined in different ways. One possible definition is the h-hop information that will be discussed in the next subsection. Again, we assume that the collection of h-hop information corresponds to one round. The number of iterations k is a constant and adjustable parameter.

B. Local solution selection

We use an extension of Wu and Li's marking process [19], called Rule K [6], in the iterative local solution (ILS). The following rule is among the most efficient (in terms of producing a small CDS) non-iterative local solutions.

Algorithm 2 Rule K as Local Solution

```
A node is unmarked if its neighbors form a clique, or are dominated by a set of connected nodes with higher priorities.
```

Formally speaking, a MANET can be represented by an undirected graph $G = (V, E)$, where V is the set of nodes and E the set of links. $N(v) = \{u|(u, v) \in E\}$ denote the neighbor set of v. Given a node set $S \subseteq V$, $N(S) = \bigcup_{v \in S} .N(v)$ is the set of nodes dominated by S. A node v can be unmarked if

1) $(u, w) \in E$ for all $u, w \in N(v)$, or
2) there exists a set of *coverage nodes* $S = \{v_1, v_2, ..., v_K\}$, such that $v_1, v_2, ..., v_K$ have higher priorities than v, the derived subgraph $G(S)$ is connected, and $N(v) - S \subseteq N(S)$.

Applying Rule K requires *h-hop information* for $h \geq 2$. By h-hop information we mean the topology and other relevant information (e.g., node priorities) collected at each node via h "Hello" message exchanges among neighbors. For each node v, its h-hop information is a subgraph $G_h(v) = (N_h(v), E_h(v))$ of the MANET. $N_h(v)$ is v's h-hop neighbor set, defined as follows: $N_0(v) = \{v\}$ and $N_h(v) = \bigcup_{u \in N(v)} N_{h-1}(u)$ for $h \geq 1$. $E_h(v)$ are links among h-hop neighbors, excluding links between two nodes that are exactly h hops away from v; that is, $E_h(v) \subseteq (N_{h-1}(v) \times N_h(v))$. The overhead for collecting h-hop information is h messages per node. Each message includes $G_{h-1}(v)$ of the current node v and is of size $O(\Delta^{h-1})$, where Δ is the maximal node degree. A small h is usually used to balance performance and overhead, such as $h = 2$ in the restricted Rule K, which incurs $O(\Delta)$ messaging cost and $O(\Delta^2)$ computing cost per node [6].

In order to use Rule K in the iterative local solution, h-hop information should also include the priorities and status of h-hop neighbors. In addition, the following restrictions are observed:

1) Initially, all nodes are considered marked.
2) At each round, only marked nodes use Rule K to determine their status, marked or unmarked, after this round. Unmarked nodes stay unmarked.
3) When applying Rule K, only marked nodes can be used as coverage nodes to unmark other marked nodes.

The resultant iterative local solution is called the *iterative Rule K*. Figure 2 shows a sample execution of iterative Rule K on a static network with 10 nodes. The restricted Rule K is used, i.e., $h = 2$. Each node is assigned a random priority at each round (iteration), which is visible to its neighbors. In round 1, three nodes with priorities 1, 3, and 4 are unmarked (represented by gray circles), because their neighbors are also neighbors of a node with a priority 6. Other nodes are marked (represented by black circles), which form a CDS.

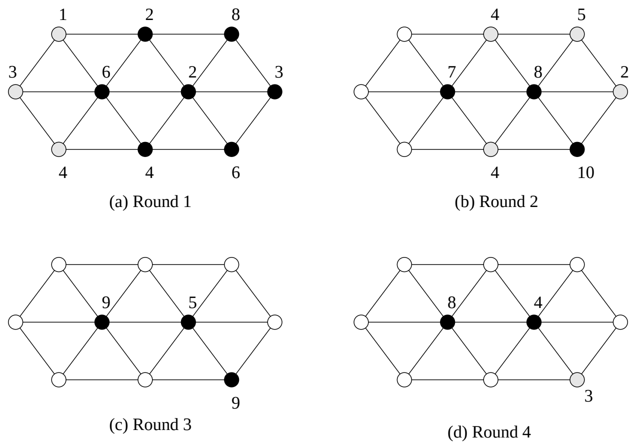


Fig. 2. The first four iterations (a-d) of iterative Rule K in a static network with 2-hop information (i.e., Rule K is restricted). Black nodes are marked (i.e., in the CDS), white nodes are unmarked, and gray nodes are newly unmarked at each round. Labels of black and gray nodes are their priorities. Priorities of white nodes are irrelevant and omitted.

In round 2, the status of three unmarked nodes (represented by white circles) is propagated to their neighbors, which will not consider them as coverage nodes in applying Rule K. According to the new priority assignments, four nodes with priorities 4, 4, 5, and 2 are unmarked. In round 3, no node is unmarked based on the new priority assignment. In round 4, however, another node with priority 3 is unmarked.

Let $V_1, V_2, \ldots, V_k$ denote the sets of marked nodes after iterations $1, 2, \ldots, k$, respectively. The following theorem shows the correctness and effectiveness of the iterative Rule K. Here we assume a static network that is connected but not completely connected.

Theorem 1: In iterative Rule K, V_i is a CDS for all $1 \leq i \leq k$, and $|V_{i+1}| \leq |V_i|$ for all $1 \leq i \leq k-1$.

After enough rounds of iteration, the marked node set is stabilized; that is, no more marked nodes can be unmarked regardless of the priority assignment.

Definition 1: An iterative local solution is stabilized at round $k^{'}$, if the set of marked nodes does not change after round $k^{'}$, i.e., $V_{k'} = V_{k'+1} = \ldots = V_k$.

In the sample network of Figure 2, the iterative Rule K is stabilized at round 4. The two marked nodes in Figure 2 (d) cannot be unmarked based on any priority assignment. When the iterative Rule K takes k' rounds to stabilize, the sets of marked nodes $V_1, V_2, \ldots, V_{k'}$ change significantly in terms of both set size and set members. The specific value of k' depends on the priority rotation scheme and network topology. For example, the number of marked nodes in Figure 2 is 7, 3, 3, and 2 in the first four rounds. A good priority assignment should achieve a fast convergence, i.e., stabilized in round $k^{'}$ with a small $k^{'}$, and converge to a small CDS as well.

C. Node priority rotation

There are several ways to rotate node priority (the corresponding scheme is called dynamic node priority). Here we denote priority as a function $p(v, i)$ of round number i and node ID v, where the IP (or MAC) address of each node can be used as its ID. To simplify the discussion, we assume that the initial priorities of n nodes are integers taken from $[1..n]$. In reality, a hash function can be used to map an IP address to an integer priority in $[1..n]$. Different nodes can have the same hash value as priority, since many local solutions (including Rule K [6]) support the same node priority case, but with less efficiency. In fact, as long as no conflict exists in the local neighborhood, efficiency will not be sacrificed.

Preposition 1: If $n = (\Delta^h)^2$ is a hash function randomly chosen from a universal class of hash functions, then the probability of a node priority collision with any node in its h-hop neighborhood is less than $1/(2\Delta^h)$.

This preposition can be easily derived from a result in [5]. Note that when h is small the condition $n = (\Delta^h)^2$ can be easily satisfied.

In the following, we will examine three possible priority rotation schemes.

- **Shifting**. Initially, the priorities $p(v, 0)$ of all nodes $v \in V$ are $1, 2, \ldots, n$, respectively. At each round i, the priority of each node v is defined as
$$p(v, i) = (p(v, i-1) + s) \ \mathbf{mod} \ n$$
where $s = \lfloor n/k \rfloor$. The pattern of the priority change can be described as a circular s-shift.
- **Shuffling**. In this scheme, node priority is changed more dramatically from round to round following the *perfect shuffle* [15] scheme. Node priority, represented as a binary string, is circularly shifted left one bit per iteration. That is,
$$p(v, i) = (p(v, i-1) \times 2) \ \mathbf{mod} \ 2^k + \lfloor p(v, i-1)/2^{k-1} \rfloor$$
Here we assume the iteration limit k satisfies $2^k \geq n$.
- **Random**. Node priority is randomly selected from $[1..n]$ at each round, i.e.,
$$p(v, i) = rnd() \ \mathbf{mod} \ n$$
where $rnd()$ is a pseudo-random number generator. Several nodes can have the same priority. The major difference between the deterministic approach (including the above shifting and shuffling schemes) and the random approach is that in the former, neighbors exchange node status (marked/unmarked) only (except the first round, where initial node priorities are also exchanged), whereas in the latter, neighbors need to exchange both node status and random node priorities generated at the current round.

The efficiency of these priority rotation schemes in ILS will be evaluated in the simulation section.

IV. Seamless Iterative Local Solution (SILS) in a Dynamic Environment

In this section, we start with a natural extension, the cyclic iterative local solution (CILS), to be used in a dynamic environment. Then we give a novel extension of the iterative local solution, the *seamless iterative local solution* (SILS).

A. Cyclic iterative local solution (CILS)

In a static environment without topology changes, the iterative local solution can produce a small CDS after k' rounds of iteration, where k' is the number of rounds needed for stabilization. In a dynamic environment with node mobility (modelled as link switched-on/off operations and node switched-on/off operations), each node must re-decide its status periodically to maintain the CDS property. A natural, but somewhat naive, extension of the iterative local solution, called *cyclic iterative local solution* (CILS), can be used to handle topology changes. In this scheme, all nodes will reset their status and the process will start over again for every k rounds of iteration. That is, all nodes are considered marked again in round $k+1$, and become gradually unmarked in the following k rounds: $k+1, k+2, \ldots, 2k$. The same process will repeat in rounds $2k+1, 2k+2, \ldots, 3k$ and so on. Figure 3 (a) shows the general pattern of the CDS size with respect to the number of rounds. Such a scheme has certain limitations and we again use Rule K to illustrate.

However, CILS suffers from the following drawbacks:

- **Broken CDS**. The cyclic scheme guarantees a CDS, V_i, for $1 \leq i \leq k$ only if there is no topology change during these k iterations. If a topology change occurs in round i, then $V_{i+1}, V_{i+2}, \ldots, V_k$ may not be a CDS. For example, if the left node with priority 6 in Figure 2 (a) switches off after round 1, the set of marked nodes in the following rounds cannot form a CDS.
- **Non-adaptiveness**. The selection of k in the cyclic scheme is non-adaptive. The cycle repeats even in a static environment. On the other hand, a large k will increase the probability of a broken CDS in a dynamic environment.

In fact, a broken CDS violates the correctness property. Non-adaptiveness destroys the progressiveness property, in which CDS should be monotonically decreasing when there is no topology change. The non-adaptiveness also makes the seamlessness property fail, since an explicit counter is needed to keep track of iteration. The nature of cyclic application also breaks the locality property because the locality property must ensure that the number and selection of marked nodes does not change significantly after each topology change. The seamless iterative local solution (SILS) discussed in the next section meets all the above properties.

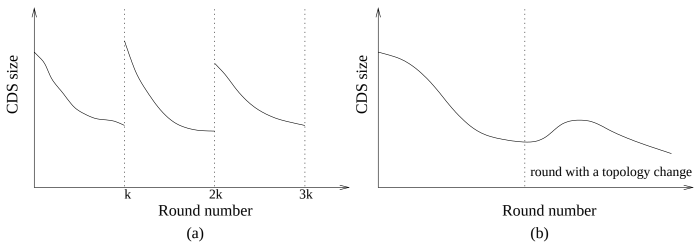


Fig. 3. Two extensions to dynamic environments: (a) cyclic iterative local solution, (b) seamless iterative local solution.

B. Seamless iterative local solution (SILS)

Again, we use Rule K to illustrate SILS, and the corresponding approach is called the *seamless iterative Rule K*. The basic idea is that the CDS formation process continues beyond k rounds of iteration. Node status (marked/unmarked) is adjusted in reaction to topology changes as the process iterates. These adjustments are conducted smoothly in small vicinities of topology changes without a global reset operation. Two important changes are made in this extension.

1) At each round, Rule K is applied at all nodes, marked or unmarked previously, to determine their new status. Note that in the original iterative solution (Section III-B), only a marked node may change its status as stated in restriction 2.
2) Node status is no longer exchanged among neighbors. The restriction 3 in the iterative Rule K (only marked nodes can be coverage nodes) is removed.

In the new extension, the original Rule K is applied based on h-hop information with a small h, including topology and priority information. At the beginning of each round i, each node collects the latest h-hop information through h rounds of "Hello" exchanges among neighbors. Each node v also selects its priority $P(v,i)$[1], which is embedded in the "Hello" messages and disseminated to its h-hop neighbors. The priority of a node can be any value that satisfies the following conditions.

1) $P(v,i) \geq P(v,i-1)$ for all $i \geq 1$.
2) $P(v,i) = P(v,i-1)$, if v is unmarked in round i.

The following theorem shows that the seamless iterative Rule K "repairs" a CDS in one round. If a topology change occurs in round $i-1$ and damages the CDS, a new CDS will be formed in round i, if no more topology changes occur in round i. This "repair rate" is the same as in the original (non-iterative) Rule K. Note that if the network topology changes in every round, no traditional local solution can maintain a CDS. Again, we use V_i to denote the set of marked nodes in round i, and assume the network is connected in each round.

[1]Here we use a upper case P to distinguish the monotonically increasing priority in SILS from the one in CILS.

Theorem 2 (Correctness): With the seamless iterative Rule K, V_i is a CDS in round i if there is no topology change in the current round.

The next theorem shows that the seamless iterative Rule K is as effective as the iterative Rule K in a static environment.

Theorem 3 (Progressiveness): With the seamless iterative Rule K, $|V_i| \geq |V_{i+1}| \geq \ldots \geq |V_j|$ if there is no topology change in rounds $i, i+1, \ldots, j$.

Finally we show that the effect of a topology change is localized. Specifically, when the seamless iterative Rule K uses h-hop information to determine the status of each node, the influence of a topology change is within $2h$ hops. We say a node v is within h hops of a topology change, if such a change can be detected by v via collecting h-hop information.

Theorem 4 (Locality): If the seamless iterative Rule K is stabilized at round i, then only nodes within $2h$ hops of a topology change may change their status after round i.

Note that the dynamic priority P integrates the treatment of both the iterative process and topology change in a seamless way. Figure 3 (b) shows a general pattern of CDS size. Notice that the CDS can increase as a response to a topology change (such as a newly switched-on node).

C. A special case

This subsection presents a special case of the seamless iterative Rule K, which is equivalent to the iterative Rule K in static networks, and has all the desirable properties of the seamless iterative Rule K in a dynamic environment. For each round $i \geq 1$, the new priority of a node v is a 2-tuple (s, p), where s is a *sequence number*, which records the most recent iteration that a node was marked. A node with a higher sequence number has a higher priority. p is a secondary priority to break a tie between two nodes with the same sequence number. Let $p(v, i)$ be one of the priority rotation schemes in Section III-A.

$$P(v,i) = \begin{cases} (i-1, p(v,i)) & : \ v \text{ is marked in round } i-1 \\ P(v, i-1) & : \ \text{otherwise} \end{cases} \tag{1}$$

All nodes are considered marked in the first round. Therefore, when $i = 1$, the corresponding priority of each node v is $P(v,1) = (0, p(v,0))$. Any new node (e.g, a node that switches on in the current round) is also considered marked. A node v added at the beginning of round i has the priority $(i-1, p(v,i))$.

Figure 4 illustrates the seamless iterative Rule K using the priority function (1). In a static network (as shown Figure 4 (a)), the marking process is stabilized after round 4. At each iteration, the dynamic Rule K produces the same set of marked nodes as in the iterative Rule K (as shown in Figure 2). Note that nodes with priority $(i-1, p)$ are unmarked after iteration i. Figure 4 (b) shows a dynamic network where a new node is added (switches on) right before round 5. After detecting this topology change, a node with priority (1,5) is marked in round 5 to maintain a CDS, while the new node with priority (4,9) is unmarked immediately. Figure 4 (c) shows the situation of node switch-off. After the topology change is detected in round 5, four unmarked nodes become marked to form a CDS. In round 6, all marked nodes adjust their priority values. A newly marked node with the lowest priority is unmarked, producing a smaller CDS.

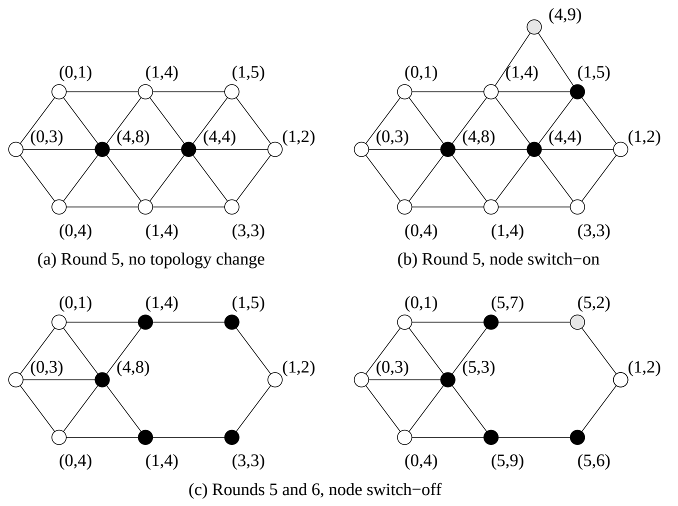


Fig. 4. A seamless iterative Rule K with 3-hop information. (a) Priority assignment after the first four rounds of iteration, where the random priority of each node in each round is the same as in Figure 2. (b) Handling a node switch-on event in round 5. (c) Handling a node switch-off event in rounds 5 and 6.

Theorem 5: In a static network, the seamless iterative Rule K using the priority function (1) produces the same set of marked nodes as the iterative Rule K at each iteration.

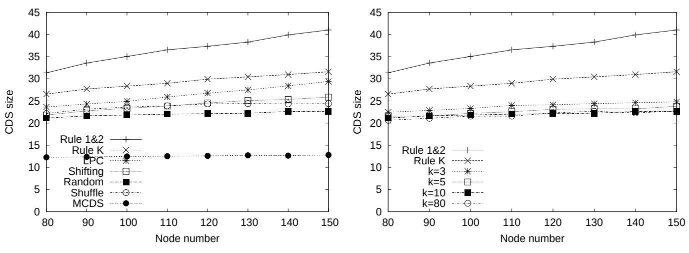
V. Simulation

This section presents results from our simulation. All algorithms are implemented on a custom simulator.

A. Static environment

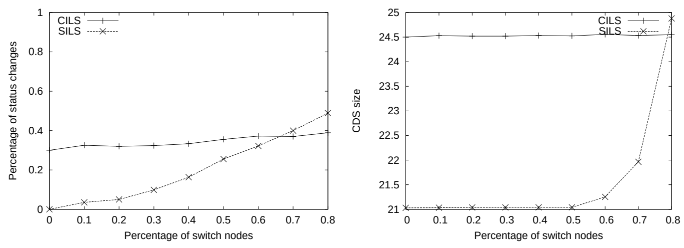
In this subsection, the performance of the proposed iterative local solution (ILS) is compared with Wu and Li's Rule 1&2 [19], Dai and Wu's Rule K [6], and the algorithm of Liu, Pan, and Cao [11] (denoted as LPC). The MCDS algorithm of Das et al [7] is a global CDS approach, which has an $O(\log \Delta)$ approximation ratio in regular graphs, where Δ is the maximum degree. We use the result of MCDS as a baseline in the comparison.

To generate a random network, n nodes are randomly placed in a restricted 1000×1000 area. The network is modeled as a unit disk graph with a fixed transmission range of 250. The tunable parameters in this simulation are as follows: (1) The node number n, which changes from 80 to 150, and (2) the



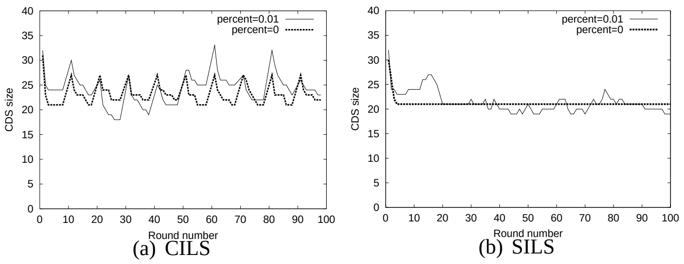
(a) Comparison of CDS size ($k = 8$) (b) Effect of round number

Fig. 5. Performance of ILS in a static environment.



(a) Average status changes (b) Average CDS size

Fig. 7. Analysis with different switched-on/off $percent$ ($n = 100, k = 10$).



(a) CILS (b) SILS

Fig. 6. CDS size at each round in the switched-on/off model ($n = 100, k = 10$).

iteration number (round number) k. The performance metric is the number of nodes in the resultant connected dominating set (CDS). We use the restricted Rule K with $h = 2$ as a sample local solution for ISL.

Figure 5 (a) shows the comparison of Rule 1&2, Rule K, LPC, and several implementations of ILS with priorities of shifting scheme (Shifting), random node value (Random), perfect shuffle (Shuffle), and MCDS, where the iteration number $k = 8$. We can see that LPC beats non-iterative Rules 1&2 and Rule K, and ILS has even smaller CDS size than LPC. Among the three node priority approaches, Random has the best performance. Shuffle is better than Shifting when the node number is relatively large. We use Random for ILS in the subsequent analysis. Figure 5 (b) shows the results of ILS with different iteration numbers (k). We can see that, with a larger k, the size of resultant CDS is smaller. But when k increases to 10, the performance can hardly be further improved. Therefore, we use $k = 10$ in the following simulation.

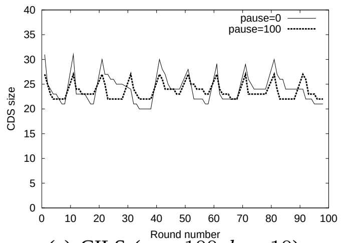
B. Dynamic environment

The switched-on/off model. CILS and SILS are evaluated in a WSN environment, where the topology change is solely caused by node switched-on/off operations. In the switched-on/off model, only a subset of deployed nodes is active. After each round, a certain percentage of active nodes switch off and the same amount of inactive nodes switch on. The simulation uses 200 deployed nodes with 100 of them active.

Figure 6 (a) demonstrates the CDS size at each round of CILS in a single run of this simulation. Two percentages of switched-on/off nodes, 0 and 0.01, are used to represent static and dynamic environments. The result is consistent with the theoretical analysis. There is little difference whether the network is static or dynamic. In each cycle, which is 10 rounds (i.e., $k = 10$), the CDS size decreases. In the next cycle, all the working nodes are marked again. Therefore, even there is no topology change, the CDS size jumps up at the beginning of each cycle. Figure 6 (b) shows the CDS size at each round of SILS in a single run. When the network is static, the CDS size decreases with rounds, achieves minimum at about round 5, and stays there. When the network is dynamic, the CDS size vibrates with rounds. But since the topology change is not significant and SILS has better locality, the vibration is less than that of CILS.

Figure 7 (a) shows the percentage of status changes per round of SILS and CILS with different switched-on/off percentages. A status change is counted whenever a node takes a different status (from marked to unmarked and vice versa) from the previous round. The number of simulation rounds is 100. The amount of average change in SILS increases with the growth of switched-on/off percentage. SILS maintains the locality property as topology changes and maintains a CDS at each round. Therefore, the more significant the network change, the larger number of status changes. CILS always has a large average status change which increases slightly with switched-on/off percentage. Figure 7 (b) shows the average CDS size of CILS and SILS with different switched-on/off percentages. We can see that when the network topology does not change or changes slightly, SILS has a smaller average CDS size than CILS; when the network changes more significantly, CILS has a smaller average CDS size, because CILS does not respond to a topology change until another cycle begins. Thus the CDS size keeps decreasing during a cycle. Note that the CDS will be broken more often in CILS in highly dynamic networks.

The random waypoint mobility model. CILS and SILS are evaluated in a random waypoint mobility model [2], where each node selects its destination randomly within the deploy-


(a) CILS ($n = 100, k = 10$)

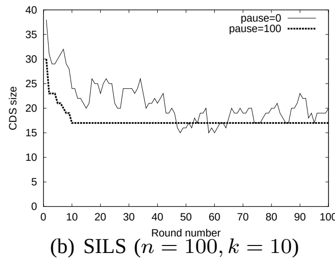
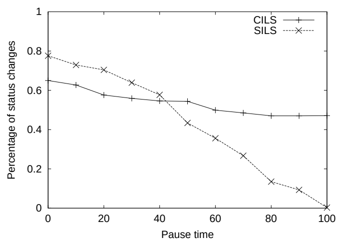

(b) SILS ($n = 100, k = 10$)

Fig. 8. CDS size at each round in the random waypoint model.


(a) Average status changes

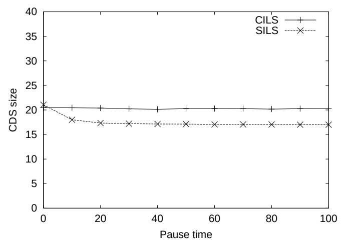

(b) Average CDS size

Fig. 9. Analysis with different $pause$ ($n = 100, k = 10$).

ment region, and moves with a speed which is uniformly chosen in $(0, V_{max}]$. When it reaches the destination, the node pauses for a time period $pause$, and then repeats this process. We set the V_{max} to be 20. $pause$ is 0 for dynamic and 100 for static, because the simulation lasts 100 rounds.

Figure 8 (a) shows the size of resultant CDS at each round of CILS, and Figure 8 (b) is that of SILS in a single run. Similar to the result shown in Figure 6, CILS has the cyclic vibration regardless of whether the nodes move or not. SILS is stabilized in round 10 when there is no movement; it vibrates when there is movement, but the vibration is calmer and the size of CDS is smaller than that of the first several rounds.

Figure 9 (a) shows the average status changes of CILS and SILS with different $pause$, and Figure 9 (b) shows the average CDS size of CILS and SILS with different $pause$, as a result of the average of 1000 simulation trials. Similar to the result shown in Figure 7, the percentage of status changes of SILS decreases with the decrease of $pause$, CILS decreases slightly. When the mobility level is relatively low, SILS has a smaller average CDS size than CILS. Under high mobility, CILS has a smaller average CDS size but the resultant CDSs during the cycle may be broken.

VI. Conclusions

This paper provides a general framework for the iterative local solution. The main contribution is the seamless integration of the iterative process and the handling of topology changes in ad hoc wireless networks which include both WSNs and MANETs. We have considered two extensions to the iterative local solution to extend its use beyond the static environment. The work of this paper provides insights on how to add some new features to a typical local solution in a dynamic environment. Some assumptions are used to ease discussion, such as the node ID as priority, synchronized "Hello" messages, the infinitely increase of sequence number, and simultaneous topology change. Those assumptions can be relaxed while preserving various desirable properties.

References

[1] K. Alzoubi, X.-Y. Li, Y. Wang, P.-J. Wan, and O. Frieder. Geometric spanners for wireless ad hoc networks. *IEEE Trans. of Parallel and Distributed Systems*, 14(5):408–421, 2003.

[2] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. *Wireless Communication & Mobile Computing (*WCMC*): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2(5):483–502, 2002.

[3] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. SPAN: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. In *Proc. of 7th ACM MOBICOM*, 2001.

[4] V. Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of Operation Research*, 4(3):233–235, 1979.

[5] T. H. Corman, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. McGraw Hill, 2001.

[6] F. Dai and J. Wu. An extended localized algorithm for connected dominating set formation in ad hoc wireless networks. *IEEE Transaction on Parallel and Distributed Systems*, 15(10):908–920, 2004.

[7] B. Das, R. Sivakumar, and V. Bharghavan. Routing in ad-hoc networks using a spine. In *Proc. of IC3N*, 1997.

[8] J. Gao, L. J. Guibas, J. Hershberger, L. Zhang, and A. Zhu. Discrete mobile centers. In *Proc. of Symposium on Computational Geometry*, 2001.

[9] D. Simplot-Ryl J. Carle. Energy-efficient area monitoring for sensor networks. *IEEE Computer*, 37(2):40–46, 2004.

[10] F. Kuhn and R. Wattenhofer. Constant-time distributed dominating set approximation. In *Proc. of 22^{nd} ACM Int. Symposium on the Principles of Distributed Computing (PODC)*, 2003.

[11] H. Liu, Y. Pan, and J. Cao. An improved distributed algorithm for connected dominating sets in wireless ad hoc networks. In *Proc. of International Symposim on Parallel and Distributed Processing and Applications (ISPA), Lecture Notes in Computer Science*, number 3358, 2004.

[12] M. V. Marathe, H. Breu, H. B. Hunt III, S. S. Ravi, and D. J. Rosenkrantz. Simple heuristics for unit disk graphs. *Networks*, 25:59–68, 1995.

[13] I. Stojmenovic. Data gathering and activity scheduling in ad hoc and sensor networks. In *Proc. of International Workshop on Theoretical Aspects of Wireless Ad Hoc, Sensor, and Peer-to-Peer Networks*, 2004.

[14] I. Stojmenovic, S. Seddigh, and J. Zunic. Dominating sets and neighbor elimination based broadcasting algorithms in wireless networks. *IEEE Trans. on Parallel and Distributed Systems*, 13(1):14–25, 2002.

[15] H. S. Stone. *High-performance computer architecture*. Addison-Wesley, 3rd edition, 1993.

[16] J. Sucec and I. Marsic. An efficient distributed network-wide broadcast algorithm for mobile ad hoc networks. CAIP Technical Report 248, Rutgers University, 2000.

[17] P. J. Wan, K. Alzoubi, and O. Frieder. Distributed construction of connected dominating set in wireless ad hoc networks. In *Proc. of IEEE INFOCOM'2002*, 2002.

[18] J. Wu and F. Dai. A generic distributed broadcast scheme in ad hoc wireless networks. In *Proc. of ICDCS'2003*, 2003.

[19] J. Wu and H. Li. On calculating connected dominating sets for efficient routing in ad hoc wireless networks. In *Proc. of ACM DIALM'99*, 1999.

[20] J. Wu and W. Lou. Forward-node-set-based broadcast in clustered mobile ad hoc networks. *Wireless Networks and Mobile Computing, a Special Issue on Algorithmic, Geometric, Graph, Combinatorial, and Vector Aspects*, 3(2):155–173, 2003.