Deterministic Dominating Set Construction in Networks with Bounded Degree*

Roy Friedman and Alex Kogan

Department of Computer Science
Technion, Israel
{roy,sakogan}@cs.technion.ac.il

Abstract. This paper considers the problem of calculating dominating sets in networks with bounded degree. In these networks, the maximal degree of any node is bounded by Δ, which is usually significantly smaller than n, the total number of nodes in the system. Such networks arise in various settings of wireless and peer-to-peer communication. A trivial approach of choosing all nodes into the dominating set yields an algorithm with the approximation ratio of $\Delta + 1$. We show that any deterministic algorithm with non-trivial approximation ratio requires $\Omega(\log^* n)$ rounds, meaning effectively that no $o(\Delta)$-approximation deterministic algorithm with a running time independent of the size of the system may ever exist. On the positive side, we show two deterministic algorithms that achieve $\log \Delta$ and $2 \log \Delta$-approximation in $O(\Delta^3 + \log^* n)$ and $O(\Delta^2 \log \Delta + \log^* n)$ time, respectively. These algorithms rely on coloring rather than node IDs to break symmetry.

1 Introduction

The *dominating set* problem is a fundamental problem in graph theory. Given a graph G, a dominating set of the graph is a set of nodes such that every node in G is either in the set or has a direct neighboring node in the set. This problem, along with its variations, such as the *connected dominating set* or the *k-dominating set*, play significant role in many distributed applications, especially in those running over networks that lack any predefined infrastructure. Examples include mobile ad-hoc networks (MANETs), wireless sensor networks (WSNs), peer-to-peer networks, etc. The main application of dominating sets in such networks is to provide a virtual infrastructure, or overlay, in order to achieve scalability and efficiency. Such overlays are mainly used to improve routing schemes, where only nodes in the set are responsible for routing messages in the network (e.g., [29, 30]). Other applications of dominating sets include efficient power management [11, 30] and clustering [3, 14].

In many cases, the network graph is such that each node has a limited number of direct neighbors. Such a limitation may result from several reasons. First,

* This work is partially supported by the Israeli Science Foundation grant 1247/09 and by the Technion Hasso Plattner Center.

M.K. Aguilera et al. (Eds.): ICDCN 2011, LNCS 6522, pp. 65–76, 2011.

it can represent a hardware limitation, such as a bounded number of communication ports in a device [8]. Second, it can be an outcome of an inherent communication protocol limitation, like in the case of BlueTooth networks composed of units, called *piconets*, that include at most eight devices [10]. Finally, performance considerations, such as space complexity and network scalability, may limit the number of nodes with which each node may communicate directly. This is a common case for structured peer-to-peer networks, where each node selects a constant number of neighbors when it joins the network [17, 25].

The problem of finding a dominating set that has a minimal number of nodes is known to be NP-complete [12], and, in fact, it is also hard for approximation [9]. Although the approximation ratio of existing solutions for the dominating set problem, $O(\log \Delta)$, was found to be the best possible (to within a lower order additive factor, unless NP has an $n^{O(\log \log n)}$-time deterministic algorithm [9]), the gap between lower and upper bounds on the running time of distributed deterministic solutions remains wide. Kuhn et al. [19] showed that any distributed approximation algorithm for the dominating set problem with a polylogarithmic approximation ratio requires at least $\Omega(\sqrt{\log n / \log \log n})$ communication rounds. Along with that, the existing distributed deterministic algorithms incur a linear (in number of nodes) running time [7, 23, 29]. This worst-case upper bound remains valid even when graphs of interest are restricted to the bounded degree case, like the ones described above.

The deterministic approximation algorithms [7, 23, 29] are based on the centralized algorithm of Guha and Khuller [13], which in turn is based on a greedy heuristic for the related set-cover problem [5]. Following the heuristic, these algorithms start with an empty dominating set and proceed as following. Each node calculates the *span*, the number of *uncovered* neighbors, including the node itself. (A node is uncovered if it is not in the dominating set and does not have any neighbor in the set.) Then it exchanges the span with all nodes within distance of 2 hops and decides whether to select itself to the dominating set based on its span and the span of nodes within distance 2. These iterations are repeated by a node v until v or at least one of its neighbors is uncovered.

The decision whether to join the dominating set in the above iterative process is taken based on the lexicographic order of the pair $\langle$ span, ID $\rangle$ [7, 23, 29]. The use of IDs to break ties leads to long dependency chains, where a node cannot join the set because of another node having higher ID. This, in turn, leads to a time complexity that is linear in the number of nodes. To see that, consider a ring, where nodes have IDs starting from 1 and increasing clockwise. At the first iteration, only the node with the highest ID $= n$ will join the set. At the second iteration, only the node with ID $= n - 3$ will join the set, since it has 3 uncovered neighbors (including itself), while nodes $n - 2$ and $n - 1$ have only 2 and 1, respectively. At the third iteration, the node with ID $= n - 6$ will join, and so on. Thus, such an approach will require roughly $n/3$ phases.

In this paper, we employ coloring to reduce the length of such dependency chains. Our approach is two-phased: we first run a coloring algorithm that assigns each node with a color, which is different from a color of any other node within distance 2. Then, we run the same iterative process described above, while using colors instead of IDs to break ties between nodes with equal span, shortening the length of the maximal chain. This approach results in a distributed deterministic algorithm with approximation ratio of $\log \Delta$ (or, more precisely, $\log \Delta + O(1)$) and running time of $O(\Delta^3 + \log^* n)$. Notice, though, that the coloring required by our algorithm can be precomputed for other purposes, e.g., time slot scheduling for the wireless channel access [15, 28]. When the coloring is given, the running time of the algorithm becomes $O(\Delta^3)$, independent of the size of the system. We also describe a modification to our algorithm that reduces its running time to $O(\Delta^2 \log \Delta + \log^* n)$ ($O(\Delta^2 \log \Delta)$ in case of coloring is already given) while the approximation ratio is increased by a constant factor.

An essential question that arises in the context of bounded degree networks is whether it is possible to construct a *local* approximation algorithm, i.e., an algorithm with a running time that depends solely on the degree bound. As have been already stated above, in the general case, Kuhn et al. [19] provide a negative answer and state that at least $\Omega(\sqrt{\log n / \log \log n})$ communication rounds are needed. Along with that, in several other related communication models, such as the *unit disc graph*, local approximation algorithms are known to exist [6]. In this paper, we show that any deterministic algorithm with a non-trivial approximation ratio requires at least $\Omega(\log^* n)$ rounds, thus answering negatively to the question stated above. In light of this lower bound, our modified algorithm leaves an additive gap of $O(\Delta^2 \log \Delta)$.

2 Related Work

Due to its importance, the dominating set problem was considered in various networking models. For general graphs, the best distributed deterministic $O(\log \Delta)$-approximation algorithms have linear running time [7, 23, 29]. In fact, these algorithms perform no better than a trivial approach where each node collects a global view of the network by exchanging messages with its neighbors and then calculates locally a dominating set approximation by running, e.g., the centralized algorithm of Guha and Khuller [13]. The only lower bound known for general graphs is due to Kuhn et. al. [19], which states that at least $\Omega(\sqrt{\log n / \log \log n})$ communication rounds are needed to find a constant or polylogarithmic approximation[1]. Their proof relies on a construction of a special family of graphs in which the maximal node degree depends on the size of the graph. Thus, this construction cannot be realized in the bounded degree model.

Another body of works considers unit-disk graphs (UDG), which are claimed to model the communication in wireless ad-hoc networks. Although the dominating set problem remains NP-hard in this model, approximation algorithms with a constant ratio are known (e.g., [6, 20]). Recently, Lenzen and Wattenhofer [22] showed that any f-approximation algorithm for the dominating set problem in the UDG model runs in $g(n)$ time, where $f(n)g(n) \in \Omega(\log^* n)$. In

[1] This work assumes unbounded local computations.

Table 1. Comparison of results on distributed deterministic $O(\log \varDelta)$-approximation of optimal dominating sets

Model	Running time	Algorithm/Lower bound
General	$\Omega(\sqrt{\log n / \log\log n})$	[19]
	$O(n)$	[7, 23, 29]
Bounded degree	$\Omega(\log^* n)$	this paper
	$O(\log^* n + \varDelta^3)$, $O(\log^* n + \varDelta^2 \log \varDelta)$	this paper

contrary, we consider a different model of graphs with bounded degree nodes, in which $\varDelta$ is not a constant number, but rather an independent parameter of the problem. This enables us to obtain more refined lower bound. Specifically, we show that while obtaining $O(\varDelta)$-approximation for the optimal dominating set in our model is possible even without any communication, any $o(\varDelta)$-approximation algorithm requires $\Omega(\log^* n)$ time. Although our proof employs a similar (ring) graph, which can be realized also in the UDG model, the formalism we use allows us to obtain our lower bound in a shorter and more straight-forward way.

The dominating set problem in bounded degree networks was considered by Chlebik and Chlebikova [4], who derive explicit lower bounds on the approximation ratios of centralized solutions. While we are not aware of any previous work on distributed approximation of dominating sets in bounded degree networks, several related problems were considered in this setting. Very recently, Astrand and Suomela et al. provided distributed deterministic approximation algorithms to a series of such problems, e.g., vertex cover [1, 2] and set cover [2]. Panconesi and Rizzi considered maximal matchings and various colorings [26].

It is worth to mention several randomized approaches that have been proposed for the general graph model and which can also be applied in the setting of networks with bounded degree. For instance, Jia et al. [16] propose an algorithm with $O(\log n \log \varDelta)$ running time, while Kuhn et. al. [21] achieve even better $O(\log^2 \varDelta)$ running time. These solutions, however, provide only probabilistic guarantees on the running time and/or approximation ratio (for example, the former achieves the approximation ratio of $O(\log \varDelta)$ in expectation and $O(\log n)$ with high probability), while our approach deterministically achieves the approximation ratio of $\log \varDelta$.

The results of previous work along with the contribution of this paper are summarized in Table 1.

3 Model and Preliminaries

We model the network as a graph $G = (V, E)$. The number of nodes is n and the degree of any node in the graph is limited by a global parameter $\varDelta$. We assume that both n and $\varDelta$ are known to any node in the system. Also, we assume that each node has a unique identifier of size $O(\log n)$. In fact, both assumptions are required only by the coloring procedure we use as a subroutine [18]. Our lower bound does not require the latter assumption and, in particular, holds for anonymous networks as well.

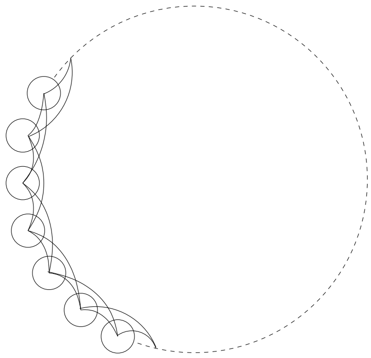

Fig. 1. A (partial) 2-ring graph $R(n, 2)$

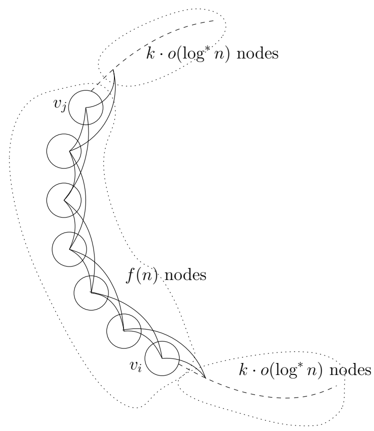

Fig. 2. A subgraph G' of $R(n, 2)$

Our model of computation is a synchronous, message-passing system (denoted as `LOCAL` in [27]) with reliable processes and reliable links. In particular, time is divided into rounds and in every round, a node may send one message of an arbitrary size to each of its direct neighbors in G, receive all messages sent to it by its direct neighbors at the same round and perform some local computation. Consequently, for any given pair of nodes v and u at distance of k edges in G, a message sent by v in round i may reach u not before round $i + k - 1$. All nodes start the computation at the same round. The time complexity of the algorithms presented below is the number of rounds from the start until the last node ceases to send messages.

Let $N_k(v, G)$ denote the k-neighborhood of a node v in a graph G, that is $N_k(v, G)$ is a set of all nodes (not including v itself), which are at most k hops from v in G. In the following definitions, all node indices are taken modulo n.

Definition 1. *A ring graph $R(n) = (V_n, E_n)$ is a circle graph consisting of n nodes where $V_n = \{v_1, v_2, ..., v_n\}$ and $E_n = \{(v_i, v_{i+1}) \mid 1 \leq i \leq n\}$. A k-ring graph $R(n, k) = (V_n, E_n^k)$ is an extension of the ring graph, where $V_n = \{v_1, v_2, ..., v_n\}$ and $E_n^k = \{(v_i, u) \mid u \in N_k(v_i, R(n)) \wedge 1 \leq i \leq n\}$.*

Notice that in $R(n, k)$ each node v has exactly $2k$ edges, one to each of its neighbors in $N_k(v, R(n))$ (see Fig. 1). Given $R(n, k)$ and two nodes $v_i, v_j \in V_n$, $i \leq j$, let $Sub(R(n, k), v_i, v_j)$ be a subgraph (V, E) where $V = \{v_k \in V_n \mid i \leq k \leq j\}$. Thus, assuming a clockwise ordering of nodes on the ring, $Sub(R(n, k), v_i, v_j)$ contains the sequence of nodes between v_i and v_j in the clockwise direction. The nodes v_i and v_j are referred to as *boundary* nodes in the sequence.

Definition 2. *Suppose A is an algorithm operating on $R(n, k)$ and assigning each node $v_i \in V_n$ a value $c(v_i) \in \{0, 1\}$. Let $r(v_i) = min_j\{j \leq i \mid \forall k, j \leq k \leq i : c(v_k) = c(v_i)\}$. Similarly, let $l(v_i) = max_j\{i \leq j \mid \forall k, i \leq k \leq j : c(v_k) = c(v_i)\}$. Then $Seq(v_i) = Sub(R(n, k), v_{r(v_i)}, v_{l(v_i)})$ is the longest sequence of nodes* *containing v_i and in which all nodes have the value $c(v_i)$. We call $v_{l(v_i)}$ as the leftmost node in $Seq(v_i)$, $v_{l(v_i)+1}$ as the second leftmost node, and so on.*

4 Proof of Bounds

4.1 Lower Bound

The minimal dominating set of any bounded degree graph has a size of at least $\frac{n}{\Delta+1}$. Thus, a simple approach for choosing all nodes of the graph into the dominating set gives a trivial $(\Delta+1)$-approximation for the optimal set. An essential question is whether a non-trivial approximation can be calculated deterministically in the bounded degree graphs in an effective way, i.e., independent of the system size. The following theorem gives a negative answer to this question.

Theorem 1. *Any distributed deterministic $o(\Delta)$-approximation algorithm for the dominating set problem in a bounded degree graph requires $\Omega(\log^* n)$ time.*

Proof. Assume, by way of contradiction, that there exists a deterministic algorithm A that finds an $o(\Delta)$-approximation in $o(\log^* n)$ time. Given a ring of n nodes, $R(n)$, the following algorithm colors it with 3 colors, for any given k.

- Construct the k-ring graph $R(n, k)$ and run A on it. For each node $v_i \in V_n$, denote the value $c(v_i)$ as 1 if A selects v into the dominating set, and as 0 otherwise.
- Every node $v_i \in V_n$ chooses its color according to whether or not v_i and some of its neighbors are chosen to the dominating set by A. Specifically, consider the sequence $Seq(v_i)$ as defined in Def. 2.
  - If v_i is not in the set, the nodes in the sequence are colored with colors 2 and 1 interchangeably. That is, the leftmost node in the sequence chooses color 2, the second leftmost node chooses color 1, the third leftmost node chooses color 2, and so on.
  - If v_i is in the set, the nodes in the sequence are colored with colors 0 and 1 interchangeably. That is, the leftmost node in the sequence chooses color 0, the second leftmost node chooses color 1, the third leftmost node chooses color 0, and so on.

The produced coloring uses 3 colors and is a subject to a straight-forward distributed implementation. Notice that the coloring is legal (i.e., no two adjacent nodes share the same color) inside sequences of nodes chosen and not chosen to the dominating set by A. Thus, the legality of the produced coloring should be verified in cases where the sequences end. Consider two neighboring nodes (in $R(n)$) v and u, where v is a left neighbor of u (i.e., v appears immediately after u in the ring when considering nodes in the clockwise direction). If v is in the set and u is not, then the color of u, being the leftmost in the sequence of nodes not in the set, is 2, while the color of v is 0 or 1. Similarly, if u is in the set and v is not, then the color of u, being the leftmost in the sequence of nodes in the set, is 0, while the color of v is 2 or 1. Thus, the produced coloring is legal.

The running time of the algorithm is $g(n) \in o(\log^* n)$ rounds spent for running A and an additional number of rounds to decide on colors. The length of the longest sequence of nodes not in the dominating set cannot exceed $2k$, since otherwise there will be a node that is not covered by any other node in the selected dominating set. Thus, the implementation of the first rule for the coloring decision requires a constant number of rounds. In the following, we show that there exists k such that the length of the longest sequence chosen to the dominating set by A is $o(\log^* n)$. Thus, for this k, nodes decide on their colors in $o(\log^* n)$ time. Thus, the running time of the algorithm to color a ring with 3 colors sums up to $o(\log^* n)$, contradicting the famous lower bound of Linial [24].

We are left with the claim that for some k, the length of the longest sequence of nodes chosen to the dominating set by A is $o(\log^* n)$. Suppose, by way of contradiction, that for any k there exists a function $f(n) \in \Omega(\log^* n)$ such that A produces a sequence of length $f(n)$. Let v_i and v_j be the boundary nodes of such a sequence s.t. $i \leq j$, and construct a subgraph $G' = Sub(R(n,k), v_{i-k\cdot g(n)}, v_{j+k\cdot g(n)})$. Notice that this subgraph contains the same $f(n)$ nodes chosen by A into the dominating set plus additional $2k \cdot g(n)$ nodes (see Fig. 2). Also note that a minimum dominating set in G', $Opt(G')$, contains $\frac{1}{2k+1}(f(n) + 2k \cdot g(n))$ nodes.

When A is run on G', nodes in the original sequence of length $f(n)$ cannot distinguish between the two graphs, i.e., $R(n,k)$ and G'. This is because in our model, a node can collect information in $o(\log^* n)$ rounds only from nodes at distance of at most $o(\log^* n)$ edges from it. Thus, being completely deterministic, A must select the same $f(n)$ nodes (plus some additional nodes to ensure that all nodes in G' are covered). Consequently, $|A(G')| \geq f(n)$, where $|A(G')|$ denotes the size of the dominating set calculated by A for the graph G'.

On the other hand, A has an $o(\Delta)$-approximation ratio, thus for any graph G, $|A(G)| \leq o(\Delta) \cdot |OPT(G)| + c$, where c is some non-negative constant. For simplicity, we will assume $c = 0$; the proof does not change much for $c > 0$. In the graph $R(n,k)$ (and G'), $\Delta = 2k$, thus there exist Δ' and k' s.t. $2o(\Delta') = 2o(2k') < 2k'+1$. In addition, since $f(n) \in \Omega(\log^* n)$ and $g(n) \in o(\log^* n)$, there exists $n' > k'$ s.t. $2k' \cdot g(n') < f(n')$.

Thus, for Δ', k' and n', we get:

$$\begin{aligned} o(\Delta') \cdot |OPT(G')| &= o(\Delta') \cdot \frac{1}{2k'+1}(f(n') + 2k' \cdot g(n')) \\ &< o(\Delta') \cdot \frac{2}{2k'+1} f(n') < f(n') \leq |A(G')|, \end{aligned}$$

contradicting the fact that A has an $o(\Delta)$-approximation ratio. □

It follows immediately from the previous theorem that no local deterministic algorithm that achieves an optimal $O(\log \Delta)$-approximation may exist.

Corollary 1. *Any distributed deterministic $O(\log \Delta)$-approximation algorithm for the dominating set problem in a bounded degree graph requires $\Omega(\log^* n)$ time.*

4.2 Upper Bound

First, we describe an algorithm that achieves $\log \Delta$-approximation in $O(\Delta^3 + \log^* n)$ time. Next, we show a modified version that runs in $O(\Delta^2 \log \Delta + \log^* n)$ time and achieves $2 \log \Delta$-approximation. We will use the following notion:

Definition 3. *A* k-distance coloring *is an assignment of colors to nodes such that any two nodes within* k *hops of each other have distinct colors.*

Our algorithm consists of two parts. The first part is a 2-distance coloring routine, implemented by means of a coloring algorithm provided by Kuhn [18]. Kuhn's distributed deterministic algorithm produces 1-distance coloring for any input graph G using $\Delta + 1$ colors in $O(\Delta + \log^* n)$ time. For our purpose, we run this algorithm on G^2 graph, created from G by (virtually) connecting each node with any of its neighbors at distance 2. This means that any message sent on such a virtual link is routed by an intermediate node to its target, increasing the running time of the algorithm by a constant factor. The second part of the algorithm is the approximation routine, which is a simple application of the greedy heuristic described in Sect. 1, where colors obtained in the first phase are used to break ties instead of IDs. That is, nodes exchange their span and color with all neighbors at distance 2 and decide to join the set if their $\langle$ span, color $\rangle$ pair is lexicographically higher than any of the received pairs.

The pseudo-code for the algorithm is given in Algorithm 1. It denotes the set of immediate neighbors of a node i by $N_1(i)$ and the set of neighbors of i at distance 2 by $N_2(i)$. Additionally, each node i uses the following local variables:

- *color*: array with values of colors assigned to each node $j \in N_2(i)$ by a 2-distance coloring routine. Initially, all values are set to $\perp$.
- *state*: array that holds the state of each node $j \in N_1(i)$. The state can be *uncovered*, *covered* or *marked*. Initially, all values are set to *uncovered*. The nodes chosen to the dominating set are those that finish the algorithm with their state set to *marked*.
- *span*: array with values for each node $j \in N_2(i)$; *span*[j] holds the number of nodes in $N_1(j) \cup \{j\}$ that are uncovered by any other node already selected into the dominating set, as reported by j. Initially, all values are set to $\perp$.
- *done*: boolean array that specifies for each node $j \in N_1(i)$ whether j has finished the algorithm. Initially, all values are set to *false*.

Theorem 2. *The algorithm in Algorithm 1 computes a dominating set with an approximation ratio of* $\log \Delta$ *in* $O(\Delta^3 + \log^* n)$ *time.*

Proof. We start by proving the bound on the running time of the algorithm. The 2-distance coloring routine requires $O(\Delta^2 + \log^* n)$ time. This is because the maximal degree of nodes in the graph G^2 is bounded by $\Delta(\Delta - 1)$ and each round of the coloring algorithm of Kuhn [18] in G^2 can be simulated by at most 2 rounds in the given graph G.

Algorithm 1. code for node i

```
1: color[i] = calc-2-dist-coloring()          ▷ use the coloring algorithm of [18]
2: distribute-and-collect(color, 2)

3: while state[j] = uncovered for any j ∈ N_1(i) ∪ {i} do
4:     span[i] := |{state[j] = uncovered | j ∈ N_1(i) ∪ {i}}|
5:     distribute-and-collect(span, 2)
6:     if ⟨span[i], color[i]⟩ > max{⟨span[j], color[j]⟩ | j ∈ N_2(i) ∧ span[j] ≠ ⊥} then
7:         state[i] := marked
8:     distribute-and-collect(state, 1)
9:     if state[j] = marked for any j ∈ N_1(i) then
10:        state[i] := covered
11:    distribute-and-collect(state, 1)
12: done
13: broadcast done to all neighbors

    distribute-and-collect(array_i , radius):
14: foreach q in [1,2, ..., radius] do
15:     broadcast array_i to all neighbors
16:     receive array_j from all j ∈ N_1(i) s.t. done[j] = false
17:     foreach node l at distance q from i do
18:         if ∃j ∈ N_1(i) s.t. done[j] = false ∧ node l at distance q − 1 from j then
19:             array_i[l] = array_j[l]
20:     done
21: done

    when done is received from j:
22: done[j] = true
23: span[j] = ⊥
```

The maximal value that the span can be assigned to is $\Delta + 1$, while the number of colors produced by the coloring procedure is $O(\Delta^2)$. Thus, the maximal number of distinct values for all ⟨ span, color ⟩ pairs is $O(\Delta^3)$. In every other iteration of the greedy heuristic (**while-do** loop in Lines 3–12 in Algorithm 1), all nodes having a maximal value of the ⟨ span, color ⟩ pair join the set. Thus, after at most $O(\Delta^3)$ iterations, all nodes are covered, while each iteration can be implemented in $O(1)$ synchronous communication rounds. Summing over both phases produces the required bound on the running time. Note that if coloring is not required, the running time is independent of n.

For the approximation ratio proof, observe that the span of a node is influenced only by its neighbors at distance of at most 2 hops. Also, notice that the dominating set problem is easily reduced to the set-cover problem (by creating a set for each node along with all its neighbors [13]). Thus, the algorithm chooses essentially exactly the same nodes as the well-known centralized greedy heuristic for the set-cover problem [5], which picks sets based on the number of uncovered elements they contain. Thus, the approximation ratio of the algorithm follows directly from the analysis of that heuristic (for details, see [5]). □

To reduce the running time of the algorithm (at the price of increasing the approximation ratio by a factor of 2), we modify the algorithm to work with an *adjusted span* for each node u. The adjusted span is the smallest power of 2 that is at least as large as the number of u's uncovered neighbors (including u itself). Thus, during the second phase of the algorithm, u exchanges its adjusted span and color with all nodes at distance 2 and decides to join the dominating set if its $\langle$ adjusted span, color $\rangle$ is lexicographically higher than that of any node at distance 2. Note that one might adjust the span to the power of any other constant $c > 1$ improving slightly the approximation ratio, but not the asymptotic running time.

Theorem 3. *The modified algorithm computes a dominating set with an approximation ratio of* $2 \log \Delta$ *in* $O(\Delta^2 \log \Delta + \log^* n)$ *time.*

Proof. The maximal value that the adjusted span can be assigned to is $\log \Delta$, while the number of colors produced by the coloring procedure is $O(\Delta^2)$. Thus, similarly to the proof of Theorem 2, we can infer that the running time is $O(\Delta^2 \log \Delta + \log^* n)$.

The factor 2 in the approximation ratio appears due to the span adjustment. In order to prove this claim, consider the centralized greedy heuristic for the set-cover problem [5] with the adjusted span modification. That is, the number of uncovered elements in a set S is replaced (adjusted) by the smallest power of 2 which is at least as large as this number, and at each step, the heuristic chooses a set that covers the largest adjusted number of uncovered elements. Following the observation in the proof of Theorem 2, setting the approximation ratio for the centralized set-cover heuristic that uses the adjusted span modification will set the proof for the approximation ratio of the modified dominating set algorithm.

When the (modified or unmodified) greedy heuristic chooses a set S, suppose that it charges each element of S by the price $1/i$, where i is the number of uncovered elements in S. As a result, the total price paid by the heuristic is exactly the number of sets it chooses, while each element is charged only once. Consider the set $S^* = \{e_k, e_{k-1}, \ldots, e_1\}$ in the optimal set-cover solution S_{opt}, and assume without loss of generality that the greedy heuristic covers the elements of S^* in the given order: $e_k, e_{k-1}, \ldots, e_1$. Consider the step at which the heuristic chooses a set that covers e_i. At the beginning of that step, at least i elements are uncovered. Thus, if the heuristic were to choose the set S^* at that step, it would pay the price of $1/i$ per element. Using the adjusted span modification, the heuristic might pay at that step at most twice the price per element covered, i.e., it pays for e_i at most $2/i$. Consequently, the total price paid by the heuristic to cover all elements in the set S^* is at most $\Sigma_{1 \le i \le k} 2/i = 2H_k$, where $H_k = \Sigma_{1 \le i \le k} 1/i = \log k + O(1)$ is the *k-th harmonic number*. Thus, since every element is in some set of S_{opt}, we get that in order to cover all elements, the modified greedy heuristic pays at most $\Sigma_{S \in S_{opt}} 2H_m = 2H_m \Sigma_{S \in S_{opt}} 1 = 2H_m |S_{opt}|$, where m is the size of the biggest set in S_{opt}. In the instance of the set-cover problem produced from the graph with the bounded degree Δ, $m = \Delta + 1$, which establishes the required approximation ratio. □

5 Conclusions

In this paper, we examined distributed deterministic solutions for the dominating set problem, one of the most important problems in graph theory, in the scope of graphs with bounded node degree. Such graphs are useful for modeling networks in many realistic settings, such as various types of wireless and peer-to-peer networks. For these graphs, we showed that no purely local, i.e., independent of the number of nodes, deterministic algorithms that calculate a non-trivial approximation may ever exist. This lower bound is complemented by two approximation algorithms. The first algorithm finds a $\log \Delta$-approximation in $O(\Delta^3 + \log^* n)$ time, while the second one achieves a $2 \log \Delta$-approximation in $O(\Delta^2 \log \Delta + \log^* n)$ time. These results compare favorably to previous deterministic algorithms with running time of $O(n)$. With regard to the lower bound, they leave an additive gap of $O(\Delta^2 \log \Delta)$ for further improvements. In the full version of this paper, we show a simple extension of our bounds for weighted bounded degree graphs.

Acknowledgments

We would like to thank Fabian Kuhn and Jukka Suomela for fruitful discussions on the subject, and to anonymous reviewers whose valuable comments helped to improve the presentation of this paper.

References

1. Åstrand, M., Floréen, P., Polishchuk, V., Rybicki, J., Suomela, J., Uitto, J.: A local 2-approximation algorithm for the vertex cover problem. In: Keidar, I. (ed.) DISC 2009. LNCS, vol. 5805, pp. 191–205. Springer, Heidelberg (2009)
2. Astrand, M., Suomela, J.: Fast distributed approximation algorithms for vertex cover and set cover in anonymous networks. In: Proc. 22nd ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), pp. 294–302 (2010)
3. Chen, Y.P., Liestman, A.L.: Approximating minimum size weakly-connected dominating sets for clustering mobile ad hoc networks. In: Proc. ACM Int. Symp. on Mob. Ad Hoc Networking and Computing (MobiHoc), pp. 165–172 (2002)
4. Chlebik, M., Chlebikova, J.: Approximation hardness of dominating set problems in bounded degree graphs. Inf. Comput. 206(11) (2008)
5. Chvatal, V.: A greedy heuristic for the set-covering problem. Mathematics of Operations Research 4(3), 233–235 (1979)
6. Czyzowicz, J., Dobrev, S., Fevens, T., Gonzalez-Aguilar, H., Kranakis, E., Opatrny, J., Urrutia, J.: Local algorithms for dominating and connected dominating sets of unit disk graphs with location aware nodes. In: Laber, E.S., Bornstein, C., Nogueira, L.T., Faria, L. (eds.) LATIN 2008. LNCS, vol. 4957, pp. 158–169. Springer, Heidelberg (2008)
7. Das, B., Bharghavan, V.: Routing in ad-hoc networks using minimum connected dominating sets. In: Proc. IEEE Int. Conf. on Comm (ICC), pp. 376–380 (1997)
8. Dong, Q., Bejerano, Y.: Building robust nomadic wireless mesh networks using directional antennas. In: Proc. IEEE INFOCOM, pp. 1624–1632 (2008)

9. Feige, U.: A threshold of $\ln n$ for approximating set cover. Journal of the ACM 45, 314–318 (1998)
10. Ferro, E., Potorti, F.: Bluetooth and Wi-Fi wireless protocols: a survey and a comparison. IEEE Wireless Communications 12(1), 12–26 (2005)
11. Friedman, R., Kogan, A.: Efficient power utilization in multi-radio wireless ad hoc networks. In: Proc. Int. Conf. on Principles of Distributed Systems (OPODIS), pp. 159–173 (2009)
12. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co. Ltd., New York (1979)
13. Guha, S., Khuller, S.: Approximation algorithms for connected dominating sets. Algorithmica 20, 374–387 (1998)
14. Han, B., Jia, W.: Clustering wireless ad hoc networks with weakly connected dominating set. Journal of Parallel and Distr. Computing 67(6), 727–737 (2007)
15. Herman, T., Tixeuil, S.: A distributed TDMA slot assignment algorithm for wireless sensor networks. In: Nikoletseas, S.E., Rolim, J.D.P. (eds.) ALGOSENSORS 2004. LNCS, vol. 3121, pp. 45–58. Springer, Heidelberg (2004)
16. Jia, L., Rajaraman, R., Suel, T.: An efficient distributed algorithm for constructing small dominating sets. In: Proc. ACM Symp. on Principles of Distr. Comp (PODC), pp. 33–42 (2001)
17. Kaashoek, M.F., Karger, D.R.: Koorde: A simple degree-optimal distributed hash table. In: Kaashoek, M.F., Stoica, I. (eds.) IPTPS 2003. LNCS, vol. 2735, pp. 98–107. Springer, Heidelberg (2003)
18. Kuhn, F.: Weak graph colorings: distributed algorithms and applications. In: Proc. Symp. on Paral. in Algorithms and Architectures (SPAA), pp. 138–144 (2009)
19. Kuhn, F., Moscibroda, T., Wattenhofer, R.: What cannot be computed locally! In: Proc. ACM Symp. on Principles of Distr. Comp. (PODC), pp. 300–309 (2004)
20. Kuhn, F., Moscibroda, T., Wattenhofer, R.: On the locality of bounded growth. In: Proc. ACM Symp. on Principles of Distr. Comp (PODC), pp. 60–68 (2005)
21. Kuhn, F., Moscibroda, T., Wattenhofer, R.: The price of being near-sighted. In: Proc. ACM-SIAM Symp. on Discrete Algorithms (SODA), pp. 980–989 (2006)
22. Lenzen, C., Wattenhofer, R.: Leveraging linial's locality limit. In: Taubenfeld, G. (ed.) DISC 2008. LNCS, vol. 5218, pp. 394–407. Springer, Heidelberg (2008)
23. Liang, B., Haas, Z.J.: Virtual backbone generation and maintenance in ad hoc network mobility management. In: Proc. IEEE INFOCOM, pp. 1293–1302 (2000)
24. Linial, N.: Locality in distributed graph algorithms. SIAM Journal on Computing 21(1), 193–201 (1992)
25. Malkhi, D., Naor, M., Ratajczak, D.: Viceroy: a scalable and dynamic emulation of the butterfly. In: Proc. ACM Symp. on Principles of Distr. Comp (PODC), pp. 183–192 (2002)
26. Panconesi, A., Rizzi, R.: Some simple distributed algorithms for sparse networks. Distributed Computing 14(2), 97–100 (2001)
27. Peleg, D.: Distributed computing: a locality-sensitive approach. SIAM, Philadelphia (2000)
28. Rhee, I., Warrier, A., Min, J., Xu, L.: DRAND: distributed randomized TDMA scheduling for wireless ad-hoc networks. In: Proc. 7th ACM Int. Symp. on Mobile Ad Hoc Networking and Computing (MobiHoc), pp. 190–201 (2006)
29. Sivakumar, R., Das, B., Bharghavan, V.: Spine routing in ad hoc networks. Cluster Computing 1(2), 237–248 (1998)
30. Wu, J., Dai, F., Gao, M., Stojmenovic, I.: On calculating power-aware connected dominating sets for efficient routing in ad hoc wireless networks. Journal of Communications and Networks, 59–70 (2002)