Initializing Newly Deployed Ad Hoc and Sensor Networks

Fabian Kuhn
Computer Engineering and Networks Laboratory
ETH Zurich, Switzerland
kuhn@tik.ee.ethz.ch

Thomas Moscibroda
Computer Engineering and Networks Laboratory
ETH Zurich, Switzerland
moscitho@tik.ee.ethz.ch

Roger Wattenhofer
Computer Engineering and Networks Laboratory
ETH Zurich, Switzerland
wattenhofer@tik.ee.ethz.ch

ABSTRACT

A newly deployed multi-hop radio network is unstructured and lacks a reliable and efficient communication scheme. In this paper, we take a step towards analyzing the problems existing during the initialization phase of ad hoc and sensor networks. Particularly, we model the network as a multi-hop quasi unit disk graph and allow nodes to wake up asynchronously at any time. Further, nodes do not feature a reliable collision detection mechanism, and they have only limited knowledge about the network topology. We show that even for this restricted model, a good clustering can be computed efficiently. Our algorithm efficiently computes an asymptotically optimal clustering. Based on this algorithm, we describe a protocol for quickly establishing synchronized sleep and listen schedule between nodes within a cluster. Additionally, we provide simulation results in a variety of settings.

Categories and Subject Descriptors: F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems

General Terms: Algorithms, Theory.

Keywords: ad-hoc networks, sensor networks, initialization, clustering, asynchronous wake-up, distributed algorithms, dominating set algorithms.

1. INTRODUCTION

In view of the great potential of ad hoc and sensor networks in a variety of application scenarios such as disaster relief, community mesh networks, monitoring and surveillance, or data gathering, it is not surprising that there has recently been a flurry of research activity in the field. Ad hoc and sensor networks are formed by autonomous nodes communicating via radio, without any additional backbone infrastructure. Typically, if two nodes are not within mutual transmission range, they communicate through intermediate nodes relaying their message, i.e., the communication infrastructure is provided by the nodes themselves. Setting up and organizing such a virtual infrastructure is an important common challenge.

During and shortly after the "big bang" (i.e., the deployment) of an ad hoc or sensor network, there is no established pattern based on which nodes could efficiently communicate. In other words, the lack of *structure* (for instance, nodes do not know their neighbors, or the number of neighbors) is characteristic of wireless multi-hop radio networks when being deployed. Before the network can actually start carrying out its intended task, the nodes must establish an efficient structure, that provides reliable point-to-point connections to higher-layer protocols and applications. The self-organized transition from an unstructured to a structured multi-hop radio network with efficient media access control (MAC) layer is called the *initialization*. The task of initializing an ad hoc and sensor network is certainly the most urgent task at hand immediately after the deployment.

In practice, the initialization problem plays a major role for various reasons. First, current initialization procedures tend to be slow. Even in a single-hop ad hoc network such as Bluetooth and for a small number of devices, the time-consumption for establishing a reasonable communication pattern is considerable. Clearly, the situation is even worse in a multi-hop scenario with large number of nodes. In wireless sensor networks [1], where the frugal usage of energy is a key issue and power control is usually an integral part of MAC layer protocols [27, 36], the initialization phase becomes even more important. Many of these protocols achieve energy savings by letting nodes sleep, i.e., turn off their radio, periodically. Until such a sleep/listen scheme is established, nodes must be awake all the time, thus dissipating valuable energy. In fact, we know of applications in which a large fraction of the node's battery is already used up during the initialization phase.

There is a multitude of excellent protocols and algorithms in literature on ad hoc and sensor networks (as well as on distributed computing in general) which would be well suited for the task of initializing and structuring newly deployed networks. Most notably, *clustering* algorithms seem to be tailor-made for the purpose of bringing structure into a chaotic network [3, 6, 7, 11, 14, 29]. Evaluating the performance of an algorithm is only sensible under a model capturing the circumstances the algorithm works under. Unfortunately, most of the above algorithms have not been primarily designed for the initialization phase, and hence, are based on model assumptions which do not closely reflect the particularly harsh conditions during the deployment.

MobiCom'04, Sept. 26-Oct. 1, 2004, Philadelphia, Pennsylvania, USA.

Many distributed algorithms are expressed under some kind of *message-passing* model. In this model, the scheduling of transmission is assumed to be handled by an existing MAC layer providing point-to-point connections between neighboring nodes. In other words, it is taken for granted that a MAC layer has previously been established on top of which the algorithm can operate. Assuming an operational MAC layer solves a variety of problems that arise in unstructured networks, such as asynchronous wake-up, collision detection, or the *hidden terminal problem*. Studying clustering in absence of an established MAC layer highlights the *chicken-and-egg* problem of the initialization phase. A MAC layer ("chicken") helps achieving a clustering ("egg"), and vice versa.[1] In a newly deployed ad-hoc/sensor network, however, there is no structure, i.e. there are neither "chickens" nor "eggs". Consequently, none of the various existing clustering algorithms based on the classic message passing model helps in the initialization process, since they basically focus on the problem of computing the "egg", given the "chicken".

Another frequent problem is that *too much knowledge* or additional hardware capability is required. Even algorithms which "merely" assume that nodes know their one or two hop neighbors are not suited for the initialization phase since the necessary discovery phase preceding the actual execution is prone to collisions and is non-trivial. Basically, if nodes knew their neighbors (or even only the number of neighbors) at the beginning of the algorithm, efficient approaches could be implemented easily. Unfortunately, in most real-world scenarios, this is not the case.

A major concern with existing algorithms and models is that they usually assume nodes to wake up or start the algorithm at the same time. In a multi-hop environment, however, it is realistic to assume that some nodes wake up (e.g. become deployed, or switched on) later than others. Due to *asynchronous wake-up*, a node cannot assume that all its neighbors have received its last transmission. New nodes may have woken up in the meantime, being completely ignorant of what has been going on in the network up to the moment. Not surprisingly, letting nodes wake up asynchronously has a big impact on almost all algorithms and naturally leads to new algorithmic designs and concepts.

The widely employed model for the study of wireless multi-hop networks is the *Unit Disk Graph* (UDG) model. The underlying assumption of this model is that nodes are placed in the plane, all of them having the same — normalized to one — transmission range. There is an edge between two nodes u and v if and only if the Euclidean distance between u and v is at most 1. The UDG model is an evident simplification of reality, since, even in a homogeneous network, the model does not account for obstacles which may obstruct signal propagation. Furthermore, it is well-known that there is no sharp threshold for the transmission (or interference) range as implied by the UDG model. A model significantly closer to reality, yet concise enough to permit stringent theoretical results, has been motivated and proposed in [5, 24]. In this paper, we will adopt this so-called *Quasi Unit Disk Graph* model (see Section 2).

Although neither the Unit Disk Graph model nor the Quasi Unit Disk Graph model is quite practical, they have generally been respected as a first step by practitioners. It is surprising, however, that in many theoretical papers in literature, even further audacious assumptions are made. In particular, it is often assumed that nodes are *distributed uniformly at random* in the plane. From a practical point of view, it is not clear whether a uniform node distribution makes sense. While even "randomly" distributed sensor networks often feature heterogenous node distributions, there are many applications scenarios where the placement of nodes is not random at all (monitoring and data gathering in a building, along a road, or within a city, for instance).

In short, in the traditional models based on which algorithms for ad hoc and sensor networks are currently analyzed, deployment specific characteristics are not addressed properly, or simply abstracted away. Algorithms, regardless of how efficient they may be in case of an established network with reliable point-to-point connections between nodes, may be of no use in a deployment scenario. This circumstance highlights the need for a common model that permits proper evaluation and comparison of protocols working during the initialization phase of ad hoc and sensor networks. On the one hand, such a model ought to be realistic enough to actually capture the characteristics of newly deployed networks. But on the other hand, we want a model which allows obtaining precise mathematical results, such that we do not need to resort solely to simulations or random distribution assumptions. In Section 2, we introduce such a model, which we believe balances the two contradicting aims.

The aim of the initialization phase is to quickly establish an efficient MAC layer. In this paper, we take a step towards this ultimate goal by giving an algorithm which efficiently computes a *clustering* of excellent quality under our restricted model. As mentioned, clustering is one prominent approach to solving the problem of bringing structure into a multi-hop radio network, and it may therefore work as an important building block when organizing a MAC layer.

Clustering allows the formation of virtual backbones, it enables efficient routing [30], it improves the usage of scarce resources, such as bandwidth and energy [15], and clustering helps realizing spatial multiplexing in non-overlapping clusters. Depending on the specific network organization problem at hand, various forms of clustering have been proposed. In this paper, we consider a clustering in which each node in the network is either a cluster-head or has a cluster-head within its communication range, such that cluster-heads can act as coordination points for the MAC scheme.

When we model a multi-hop radio network as a graph $G = (V, E)$, clustering can be formulated as a classic graph theory problem: In a graph, a *dominating set* is a subset of nodes such that for every node v, either a) v is in the dominating set or b) a direct neighbor of v is in the dominating set. As it is advantageous to compute a dominating set with few *dominators* (i.e. cluster heads), we study the well known *minimum dominating set* (MDS) problem which asks for

[1] In a clustering based on *dominating sets*, for instance, we could obtain a coloring as follows: Starting with a connected dominating set — which can be computed from a dominating set [33] —, the cluster-heads choose colors, so that no two cluster-heads within a certain range have the same color. If this range is chosen to be large enough and a color is associated with a frequency, no two neighboring clusters interfere. Reversely, we can derive a maximal independent set (MIS) from a coloring of the cluster-heads by first choosing all nodes of color 1, and then adding as many nodes not violating independence as possible for subsequent colors. In a Unit Disk Graph, a MIS is a constant approximation for a minimum dominating set[32].

a dominating set of minimum size. Once the dominators are chosen by the algorithm, non-dominator nodes can be clustered around dominators in various, application-specific ways. For example, non-dominators may associate with the dominator providing the strongest signal, or it may associate with all dominators within its transmission range. Since our algorithm is guaranteed to produce at most a constant number of dominators within the transmission range of any given node, even this latter procedure is feasible from the point of view of energy efficiency.

In this paper, we propose a clustering algorithm which is explicitly defined for being used during the initialization of unstructured multi-hop radio networks lacking any kind of a-priori structure or MAC layer. In spite of the vast literature on dominating set algorithms, this is — to the best of our knowledge — the first algorithm addressing the requirements of newly deployed networks. Even under our "harsh" initialization model, the algorithm computes an asymptotically optimal clustering in polylogarithmic time. Capable of working in total absence of any MAC layer, we believe that the algorithm has practical relevance in scenarios in which traditional dominating set algorithms fail.

Based on this clustering algorithm, we briefly discuss a protocol for setting up periodic sleep/listen schedules within clusters in a quick and energy-efficient way. As mentioned above, organizing such schedules is of practical importance in many sensor network applications, and we believe that our protocol is a first step towards a solution.

The paper is organized as follows. Section 2 introduces the model for the initialization phase. After introducing notations in Section 3, the algorithm is developed and analyzed using multiple communication channels in Sections 4 and 5. The subsequent Section 6 extends our analysis to the single-channel case. Simulation results are presented in Section 8. The technique of simulating several independent channels with a single channel may be of independent interest. An overview over related work is given in Section 9, before the paper is concluded in Section 10.

2. MODEL

In this section we introduce our model which will be used throughout the rest of this paper. We believe that the model is on the one hand strong enough to derive mathematically precise results, but on the other hand close enough to reality to actually have practical merit. In particular, we model the conditions of an ad hoc or sensor network after its deployment by making the following assumptions.

- We model the network as a *Quasi Unit Disk Graph* (QUDG) [5, 24]. In a d-QUDG, two nodes are connected if their Euclidean distance is less than or equal to d, d being a parameter between 0 and 1. Furthermore, if the distance between two nodes is greater than 1, there is no edge between them. In the range between d and 1, the existence of an edge is unspecified. Such an edge may or may not be available. The signal propagation of omnidirectional antennas does not form a clear-cut disk. Up to a certain distance (modelled with the parameter d), the signal is strong enough to (almost) completely ensure communication. Beyond a certain distance (normalized to 1), on the other hand, reception is impossible due to physical constraints. In the area in between these two regions, a transmission may or may not be successfully received. Observe that the QUDG model is equivalent to the traditional unit disk graph model when setting $d := 1$. Further, note that reasonable values for d are constants and independent of the number of nodes N in the network.

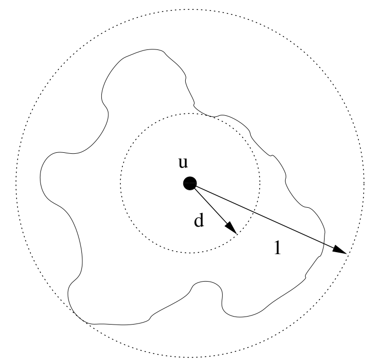

Figure 1: The transmission range of node u varying between d and 1.

- In most ad hoc and sensor networks, there exist nodes that are not within mutual transmission range. We consider such wireless *multi-hop* networks. The multi-hop nature of such networks entails a variety of additional problems. Some of the neighbors of a sending node may receive a transmission, while others are experiencing interference from other senders and do not receive the transmission. Further, the *hidden terminal problem* complicates building a maximal independent set on which several proposed dominating set algorithms are based [2, 32, 33].

- Nodes do not feature a reliable *collision detection* mechanism. That is, nodes are not capable of distinguishing between the situation in which two or more neighbors are sending and the situation in which no neighbor is sending. Further, a sending node itself does not have a collision detection mechanism either, it does not know how many (if any at all!) neighbors have received its transmission correctly. It has been argued that not assuming any collision detection mechanism is realistic from a practical point of view in many scenarios. Nodes may be tiny sensors in a sensor network where equipment is restricted to the minimum due to limitations in energy consumption, weight, or cost [34]. Naturally, algorithms without collision detection are less efficient than algorithms with collision detection. Note that the absence of a reliable collision detection mechanism prevents us from using protocols such as Busy Tone Multiple Access (BTMA) [31] or Dual Busy Tone Multiple Access (DBTMA) [8].

- Nodes can wake up *asynchronously* at any time and consequently, they do not have access to a global clock. Due to asynchronous wake-up, some nodes may still be sleeping, while others are already transmitting. Therefore, at any moment in time, there may be sleeping nodes which do not participate in the communication in spite of their being within the transmission range of a sending node. Sleeping nodes can neither send nor receive any messages, and they are not woken up by messages sent by neighboring nodes. In a multi-hop environment, it is realistic to assume that some nodes wake up (e.g. become deployed, or switched on) later than others. It is important to observe the implications of asynchronous wake-up. If all nodes started the algorithm simultaneously, we could easily assume an ALOHA kind of MAC-layer in which each node sends with probability $\Theta(1/n)$. It is well known that this approach leads to a quick and simple communication scheme on top of which existing dominating set algorithms can be used. If nodes wake up asynchronously, however, the same approach results in an expected linear runtime if only one single node wakes up for a long time. More sophisticated algorithms are required to guarantee polylogarithmic runtime in case of asynchronous wake-up.

- We assume that nodes have no knowledge about the other nodes' distribution or wake-up pattern. Particularly, nodes are completely clueless about the number of nodes in their neighborhood. The only knowledge a-priori given to the nodes is an upper bound N for the total number of nodes $n = |V|$ in the graph. While n is unknown, all nodes have the same estimate $N \geq n$. As proved in [18], it is impossible to construct efficient algorithms if nodes wake up asynchronously and do not have access to a global clock without having any estimate of n. More precisely, it was proved that at least $\Omega(n/\log n)$ time-slots are required before even *one single message* can be transmitted without collision in such a scenario. In order to circumvent this unsatisfactory, almost linear running time, an estimate N for n is inevitable. In practice, the number of nodes in a network may not be known exactly, but it can roughly be estimated in advance. Optionally, the algorithm's runtime may be improved by giving all nodes an additional upper bound Δ for δ, δ denoting the maximum degree (number of neighbors) in the network.

- Finally, for reasons mentioned in the introduction, we do not assume any kind of random node distribution. Instead, we aim for results holding for *every possible placement* of nodes.

Working precisely under this model, the algorithm in Section 4 computes (within polylogarithmic time) a dominating set which is only a constant factor larger than the optimum. Being able to obtain "hard" provable results for this algorithm serves as an example of the models conciseness.

3. NOTATION

In this Section, we formally introduce some technicalities and notations used in the sequel of the paper.

A node is called *sleeping* before its wake-up, and *active* thereafter. Sleeping nodes can neither send nor receive any messages. In Sections 4 and 5, we assume that nodes have three independent communication channels Γ_1, Γ_2, and Γ_3. These independent channels may be realized using a frequency division multiple access (FDMA) scheme. Having three communication channels simplifies the analysis, but it is not an indispensable necessity to obtain our results. We prove in Section 6 that the same approximation-ratio can be achieved even with a single communication channel in polylogarithmic time.

For the sake of simplicity, we assume — for the analysis of the algorithm — that time is divided into time-slots. However, we attach importance to the observation that our algorithm *does not rely on synchronized time-slots* in any way. Since nodes do not have access to a global clock and synchronizing time-slots is an expensive task, such an assumption would be highly unrealistic. In this paper, it is solely for the purpose of analyzing the algorithm that we assume slotted channels. This simplification of the analysis is justified due to the standard trick which has been introduced in the analysis of slotted vs. unslotted ALOHA [28]. In [28], it is shown that the realistic unslotted case and the idealized slotted case differ only by a factor of 2. The basic intuition is that a single packet can cause interference in no more than two successive time-slots. Similarly, by analyzing our algorithm in an "ideal" setting with synchronized time-slots, we obtain a result which is only by a factor 2 better as compared to the more realistic unslotted setting.

In each time-slot, a node can either send or not send. A node successfully receives a message in a time-slot if and only if *exactly* one node in its neighborhood has sent a message in the same time-slot. The variable p_k denotes the probability that node k sends a message in a given time-slot on channel Γ_1. The term *sum of sending probabilities* refers to the sum of sending probabilities on channel Γ_1.

4. ALGORITHM

The main idea of the algorithm is that nodes, after some initial waiting, compete to become dominators by exponentially increasing their sending probability on Γ_1. Note that in light of asynchronous wake-up, this exponential increase is indispensable in order to achieve sublinear running time. Channels Γ_2 and Γ_3 are then used to guarantee that the number of further dominators emerging in the neighborhood of an already existing dominator remains small.

Each node starts executing the quasi unit disk graph clustering algorithm (Algorithm 1) immediately upon waking up. In case the algorithm is invoked without the additional estimate Δ for δ, N simply serves as an upper bound for the node degree $\Delta := N$. As shown in Theorem 4.2, the algorithm's runtime decreases slightly if Δ is given as part of the input.

The algorithm starts with a waiting phase (lines 4 to 8) in which a newly awakened node waits for messages on all three channels without sending itself. Intuitively, nodes which are waking up late should not interfere with already existing dominators. Before actively trying to become a dominator, a node first listens for existing dominators in their neighborhood. More specifically, we will choose the parameter α as to ensure that a awakening node, which is already within an existing dominators transmission range, does not become dominator itself.

The competition phase starting in line 9 succeeds the waiting phase. Nodes not having received any messages from a dominator during the waiting phase will now try to compete for becoming a dominator themselves. Technically, the competition phase consists of $\log \Delta + 1$ rounds, each of which contains $\alpha \cdot \lceil \log N/d^2 \rceil$ time-slots. As described in Section 2, these time-slots are not required to be synchronized between the nodes. In every time-slot, a node sends with probability p on channel Γ_1. Starting from a small value, this

Algorithm 1 QUDG Clustering Algorithm

decided := false;
dominator := false;
upon wake-up do:
1: **if** Δ not given as input **then**
2:   $\Delta := N$;
3: **fi**
4: **for** $s := 1$ to $\alpha \cdot \lceil \log^2 N/(d^2 \log\log N) \rceil$ **do**
5:   **if** message **received then**
6:     decided := true;
7:   **fi**
8: **od**
9: **for** $r := 0$ to $\lceil \log \Delta \rceil$ **do**
10:   **for** $s := 1$ to $\alpha \cdot \lceil \log N/d^2 \rceil$ **do**
11:     $\gamma_1 := 0$; $\gamma_2 := 0$; $\gamma_3 := 0$;
12:     **if not** decided **then**
13:       $\gamma_1 := 1$, with probability $p := \eta d^2 2^{-\lceil \log \Delta \rceil + r}$;
14:       **if** $\gamma_1 = 1$ **then**
15:         dominator := true;
16:       **else if** message **received then**
17:         decided := true;
18:       **fi**
19:     **fi**
20:     **if** dominator **then**
21:       $\gamma_2 := 1$, with probability $\eta d^2 \log\log N/\log N$;
22:       $\gamma_3 := 1$, with probability $\eta d^2 \log\log N/\log^2 N$;
23:     **fi**
24:     **for** $c := 1$ to 3 **do**
25:       **if** $\gamma_c = 1$ **then**
26:         **send** on channel Γ_c
27:       **fi**
28:     **od**
29:   **od**
30: **od**
31: **if not** decided **then**
32:   dominator := true;
33:   decided := true;
34: **fi**
35: **if** dominator **then**
36:   **loop**
37:     **send** on Γ_2 , with prob. $\eta d^2 \log\log N/\log N$;
38:     **send** on Γ_3 , with prob. $\eta d^2 \log\log N/\log^2 N$;
39:   **end loop**
40: **fi**

sending probability p is doubled in every round. A node becomes a dominator when sending its first message on channel Γ_1. After becoming a dominator, a node starts sending on channels Γ_2 and Γ_3 with probability $\eta d^2 \log\log N/\log N$ and $\eta d^2 \log\log N/\log^2 N$, respectively, in addition to its sending on Γ_1 with probability p. Once a node becomes a dominator, it will remain so for the rest of the algorithm.

A key observation is that we have to prevent the sum of sending probabilities on channel Γ_1 of all nodes in a neighborhood from reaching too high values. Otherwise, too many collisions will occur, leading to a large number of dominators. Therefore, upon receiving its first message (without collision) on any of the channels, a node becomes *decided* and stops sending on Γ_1. A node v being decided means that v knows of the existence of a dominator in its neighborhood. Consequently, such a node v stops competing to become dominator itself by sending on Γ_1. The parameters η and α of the algorithm are defined as follows:

$$\alpha := \lceil \log^{-1}(753/752) \rceil \qquad \eta := 2^{-7}$$

The parameter α is chosen large enough to ensure that with high probability, there is a round in which one competing node will send without collision. Once this happens, all other (awake) nodes in the vicinity of the sending nodes will know about their being covered. Finally, the choice of η maximizes the probability of a successful execution of the algorithm. The exact values of η and α are determined by the necessity that all claims in Section 5 hold with high probability.

It remains to give an intuitive explanation why the sending probabilities on channels Γ_2 and Γ_3 are chosen as in the algorithm. Basically, the idea is that a newly awakened node v should received a message during the first $\alpha \cdot \lceil \log^2 N/(d^2 \log\log N) \rceil$ waiting slots in case it is already dominated. The number of dominators in v's vicinity being unknown, we have to make sure that v receives a message regardless of how many dominators have previously been chosen. It turns out that for that purpose, two communication channels are necessary (see Lemma 5.10). Defining the sending probabilities too small or too large could lead to the undesirable situation in which nodes become dominator in spite of their being covered at the time of their waking up.

Correctness of the algorithm and time-complexity (defined as the number of time-slots of a node between wake-up and decision) follow immediately:

Theorem 4.1. *The algorithm computes a correct dominating set.*

Proof. Every node which has not received a message from a dominator at the end of the algorithm will decide to become a dominator in line 28. Hence, every node is either dominator or has a dominator in its neighborhood. □

Theorem 4.2. *Every node decides whether or not to become dominator in time*

$$O\left(\frac{\log N}{d^2}\left(\log \Delta + \frac{\log N}{\log\log N}\right)\right)$$

Proof. The number of iterations in the first for-loop is $\alpha \cdot \lceil \log^2 N/(d^2 \log\log N) \rceil$. The two nested loops of the algorithm are executed $\lceil \log \Delta \rceil + 1$ and $\alpha \cdot \lceil \log N/d^2 \rceil$ times, respectively. At the end of these two loops, all remaining undecided nodes decide to become dominator. The time-complexity now follows from α being a constant. □

Remark 1: For all reasonable (constant) values of d, the time-complexity given in Theorem 4.2 reduces to

$$O\left(\frac{\log^2 N}{\log\log N}\right) \qquad \text{for} \qquad 1 \leq \Delta \leq N^{1/\log\log N},$$
$$O(\log N \log \Delta) \qquad \text{for} \qquad N^{1/\log\log N} \leq \Delta \leq N.$$

Remark 2: Note that the upper bounds N and Δ do not have to be particularly tight in order to obtain a good time-complexity. If, for instance, $N \leq n^\lambda$ and $\Delta \leq \delta^\lambda$ for a given $\lambda \geq 1$, the time-complexity only increases to $O\left(\frac{\lambda^2 \log N}{d^2}\left(\log \Delta + \frac{\log N}{\log\log N}\right)\right)$.

5. ANALYSIS

This section contains the main theoretical contribution of this paper. It shows that the expected number of dominators in the network is within $\mathrm{O}(1/d^2)$ of an optimal solution, which reduces to $\mathrm{O}(1)$ for any constant value of d. As mentioned in Section 3, we can simplify the analysis by assuming all nodes operate with synchronized time-slots, such that the slot boundaries are perfectly aligned and a single packet transmission may cause a collision in exactly one time-slot. This idealized analysis yields a result which is better only by a factor 2 as compared to the realistic unsynchronized setting.

We start the section by providing two facts, the first of which has been proven in [18] and the second can be found in standard mathematical textbooks.

FACT 5.1. *Given a set of probabilities $p_1 \dots p_n$ with $\forall i : p_i \in [0, \frac{1}{2}]$, the following inequalities hold:*

$$\left(\frac{1}{4}\right)^{\sum_{k=1}^{n} p_k} \leq \prod_{k=1}^{n} (1-p_k) \leq \left(\frac{1}{e}\right)^{\sum_{k=1}^{n} p_k}.$$

FACT 5.2. *For all n, t, such that $n \geq 1$ and $|t| \leq n$,*

$$e^t \left(1 - \frac{t^2}{n}\right) \leq \left(1 + \frac{t}{n}\right)^n \leq e^t.$$

In order to analyze the performance of the algorithm, we cover the plane with imaginary circles C_i of radius $r = d/2$ by placing them on an hexagonal grid as shown in Figure 2. Let D_i be the circle centered at the center of C_i having radius $R = 1 + d/2$. It can be seen that D_i is fully or partially covering a number of smaller circles C_j. In the analysis, we will continuously make use of the observation that every node in a circle C_i can hear all other nodes in C_i. On the other hand, nodes outside D_i are not able to cause a collision in C_i. The following lemma bounds the number of circles covered by D_i.

LEMMA 5.3. *Circle D_i is covering at most $5/d^2 + 15/d + 11$ circles C_i.*

PROOF. Letting χ be the smallest number of disks of radius r needed to cover a disk D_i, the limit of the ratio of the area of D_i to the area of the smaller disks is $\frac{3\sqrt{3}}{2\pi}$ [20]. All circles intersecting D_i are completely inside the area D'_i, where D'_i has radius $R' := R + 2r = 1 + \frac{3d}{2}$. Hence, we can write

$$\frac{\left(1 + \frac{3d}{2}\right)^2 \pi}{\chi \cdot \left(\frac{d}{2}\right)^2 \pi} \geq \frac{3\sqrt{3}}{2\pi}$$

and by solving for χ, we obtain χ

$$\chi \leq \frac{3\sqrt{3}}{2\pi} \cdot \left(\frac{1}{d^2} + \frac{3}{d} + \frac{9}{4}\right).$$

□

We first give a broad outline of the proof which contains four major steps, from Lemma 5.6 to Lemma 5.10. First, we bound the sum of sending probabilities in a circle C_i. In a second step, this gives us an upper bound on the number of collisions in a circle before at least one dominator emerges. Thirdly, we give a probabilistic bound on the number of sending nodes per collision. In the last step, we show that nodes waking up late in an already covered circle do not become dominator. More specifically, we show that all these claims hold with high probability.

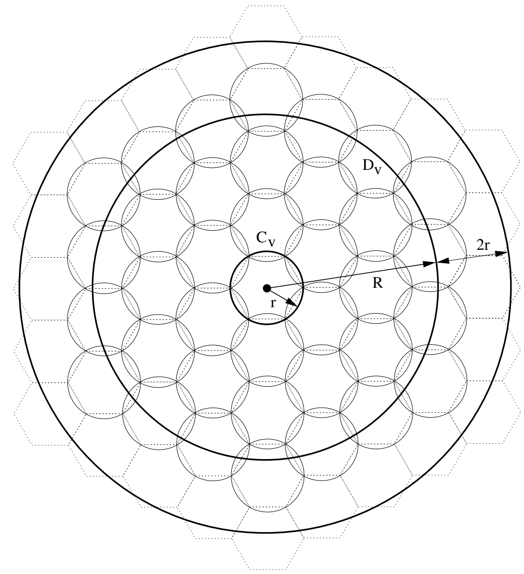

Figure 2: Circles C_i and D_i with radii $r = d/2$ and $R = 1{+}d/2$. All C_i are entirely within circle D'_i having radius $R' = 1 + 3d/2$

While the quality of the upper bounds N and Δ does influence the algorithm's running time, it does not affect its *approximation ratio*. The following lemma shows that it is sufficient to assume $N = n$ and $\Delta = \delta$ in order to derive an upper bound for the expected number of dominators.

LEMMA 5.4. *The expected number of dominators in the case $n < N$ or $\delta < \Delta$ is at most the expected number of dominators in the case $n = N$ and $\delta = \Delta$.*

PROOF. Assume for contradiction that $n_1 < N$ leads to more dominators than $n_2 = N$. Since the adversary controls the wake-up schedule of all nodes, it can simply let $n_2 - n_1$ nodes sleep infinitely long, such that, the two cases are indistinguishable. For that reason, the expected number of dominators must be equal in both cases, which contradicts the assumption. The same argument holds for $\delta = \Delta$, too. □

For the sake of readability, we will use the notation n and Δ to denote the values $N = n$ and $\Delta = \delta$ for the rest of the analysis.

When bounding the sum of sending probabilities in the first step, we are interested in time-slots in which exactly one nodes sends.

DEFINITION 5.1. *Consider a circle C_i. Let t be a time-slot in which a message is sent by a node $v \in C_i$ on channel Γ_1 and received (without collision) by all other nodes in C_i. We say that circle C_i* clears *itself in time-slot t. Let t_0 be the first such time-slot. We say that circle C_i* terminates *itself in time-slot t_0. For all time-slots $t' \geq t_0$, we call C_i* terminated.

Further, we are interested in time-slots in which the sum of sending probabilities exceeds a certain threshold.

DEFINITION 5.2. *Let $p_k(t)$ be the sending probability of node k on channel Γ_1 in time-slot t. We define the time slot t_i^j such that, for the j^{th} time in C_i, the sum of sending probabilities in C_i exceeds the threshold ηd^2. Formally, for each such time slot t_i^j, it holds that*

$$\sum_{k\in C_i} p_k(t_i^j-1) < \eta d^2 \quad and \quad \sum_{k\in C_i} p_k(t_i^j) \geq \eta d^2.$$

In other words, t_i^0 is the time-slot in which the sum of sending probabilities in C_i exceeds ηd^2 for the first time and t_i^j is the time-slot in which this threshold is surpassed for the j^{th} time in C_i. The following lemma bounds the sum of sending probabilities in a circle.

LEMMA 5.5. *Let φ be $\alpha \cdot \lceil \log n/d^2 \rceil$. For all time-slots $t' \in [t_i^j \dots t_i^j + \varphi - 1]$, the sum of sending probabilities in C_i is bounded by*

$$\sum_{k\in C_i} p_k \;\leq\; 3\eta d^2.$$

PROOF. According to the definition of t_i^j, the sum of sending probabilities $\sum_{k\in C_i} p_k$ at time $t_i^j - 1$ is less than ηd^2. By the definition of Algorithm 1, all nodes which are active at time t_i^j will double their sending probability p_k exactly once in the following $\alpha \cdot \lceil \log N/d^2 \rceil$ time-slots. Previously inactive nodes may wake up during that interval. There are at most δ of such newly active nodes and each of them will send with the initial sending probability $\frac{\eta d^2}{2^{\log \Delta}} = \eta d^2/\Delta$ in the given interval. In $[t_i^j \dots t_i^j + \varphi - 1]$, we get

$$\begin{aligned}\sum_{k\in C_i} p_k \quad &\leq \quad 2\eta d^2 + \sum_{k\in C_i} \frac{\eta d^2}{\Delta} \\ &\leq \quad 2\eta d^2 + \frac{d^2 \eta \delta}{\Delta} \;\leq\; 3\eta d^2.\end{aligned}$$

□

Using the above lemma, we can formulate a probabilistic bound on the sum of sending probabilities in a circle C_i. Intuitively, we show that before the bound can be surpassed, C_i does either clear itself or some nodes in C_i become decided such that the sum of sending probabilities decreases.

LEMMA 5.6. *The sum of sending probabilities of nodes in a circle C_i is bounded by*

$$\sum_{k\in C_i} p_k \;\leq\; 3\eta d^2$$

with probability at least $1 - \mathrm{o}\left(\frac{1}{n^2}\right)$. The bound holds for all C_i in G with probability at least $1 - \mathrm{o}\left(\frac{1}{n}\right)$.

PROOF. Again, let φ denote $\alpha \cdot \lceil \log n/d^2 \rceil$. The proof is by induction over all time-slots t_i^j in ascending order. Let $t' := t_i^0$ be the very first such time-slot in the network. Lemma 5.5 states that the sum of sending probabilities in C_i is bounded by $3\eta d^2$ in the interval $[t' \dots t' + \varphi - 1]$. We now show that in this interval, the circle C_i either clears itself or the sum of sending probabilities falls back below ηd^2 with high probability.

If some of the active nodes in C_i receive a message from a neighboring node, the sum of sending probabilities may fall back below ηd^2. In this case, the sum does obviously not exceed $3\eta d^2$.

If the sum of sending probabilities *does not fall back* below ηd^2, the following two inequalities hold for the duration of the interval $[t' \dots t' + \varphi - 1]$:

$$\eta d^2 \;\leq\; \sum_{k\in C_i} p_k \;\leq\; 3\eta d^2 \quad : \quad in\ C_i \tag{1}$$

$$0 \;\leq\; \sum_{k\in C_j} p_k \;\leq\; 3\eta d^2 \quad : \quad in\ C_j \in D_i, i \neq j. \tag{2}$$

The second inequality holds because t' is the very first time-slot in which the sum of sending probabilities exceeds ηd^2. Hence, in each $C_j \in D_i$, the sum of sending probabilities is at most $3\eta d^2$ in the interval $[t' \dots t' + \varphi - 1]$. (Otherwise, one of these circles would have reached ηd^2 before circle C_i and t' is not the first time-slot considered).

We will now compute the probability that C_i clears itself within the interval $[t' \dots t' + \varphi - 1]$. Circle C_i clears itself when exactly one node in C_i sends and no other node in $D_i \setminus C_i$ sends. The probability P_0 that no node in any neighboring circle $C_j \in D_i, j \neq i$ sends is

$$\begin{aligned} P_0 \quad &= \quad \prod_{\substack{C_j\in D_i\\ j\neq i}} \prod_{k\in C_j} (1-p_k) \\ &\underset{\text{Fact 5.1}}{\geq} \quad \prod_{\substack{C_j\in D_i\\ j\neq i}} \left(\frac{1}{4}\right)^{\sum_{k\in C_j} p_k} \\ &\underset{\text{Lemma 5.5}}{\geq} \quad \prod_{\substack{C_j\in D_i\\ j\neq i}} \left(\frac{1}{4}\right)^{3\eta d^2} \;\geq\; \left[\left(\frac{1}{4}\right)^{3\eta d^2}\right]^{\frac{5}{d^2}+\frac{15}{d}+11} \\ &\geq \quad \left(\frac{1}{4}\right)^{\eta(15+45d+33d^2)} \;>\; \left(\frac{1}{4}\right)^{\frac{3}{4}}. \end{aligned} \tag{3}$$

Let P_{suc} be the probability that exactly one node in C_i sends:

$$\begin{aligned} P_{suc} \quad &= \quad \sum_{k\in C_i} \left(p_k \cdot \prod_{\substack{l\in C_i\\ l\neq k}} (1-p_l)\right) \\ &\geq \quad \sum_{k\in C_i} p_k \cdot \prod_{l\in C_i} (1-p_l) \\ &\underset{\text{Fact 5.1}}{\geq} \quad \sum_{k\in C_i} p_k \cdot \left(\frac{1}{4}\right)^{\sum_{k\in C_i} p_k} \qquad (4) \\ &\geq \quad \eta d^2 \cdot \left(\frac{1}{4}\right)^{\eta d^2}. \end{aligned}$$

The last inequality holds because function (4) is strictly increasing in $[\eta d^2, 3\eta d^2]$.

The probability P_c that exactly one node in C_i and no other node in D_i sends is therefore given by

$$P_c \;=\; P_0 \cdot P_{suc} \;\geq\; \eta d^2 \left(\frac{1}{4}\right)^{\eta d^2 + \frac{3}{4}}.$$

P_c is a lower bound for the probability that C_i clears itself in a time-slot $t \in [t' \dots t' + \varphi - 1]$.

Using abbreviations $k_1 := \eta\left(\frac{1}{4}\right)^{3/4}$ and $k_2 := \left(\frac{1}{4}\right)^{\eta}$, we can write the probability $\overline{P_{term}}$ that circle C_i does *not clear* itself during the entire interval of length φ as,

$$
\begin{aligned}
\overline{P_{term}} &\leq \left(1-\eta d^2\left(\frac{1}{4}\right)^{\eta d^2+\frac{3}{4}}\right)^{\alpha\cdot\lceil\log n/d^2\rceil} \\
&\leq \left[\left(1-k_1 d^2 k_2^{d^2}\right)^{(1/d^2)}\right]^{\alpha\log n} \\
&\leq \left[\left(1-k_1 d^2 k_2\right)^{(1/d^2)}\right]^{\alpha\log n} \\
&\leq e^{-k_1k_2k_3\log n} \leq n^{-k_1k_2\alpha/\ln 2} \in \mathrm{o}\big(n^{-2}\big)\,.
\end{aligned}
$$

We have thus shown that within the interval $[t' \ldots t'+\varphi-1]$, the sum of sending probabilities in C_i either falls back below ηd^2 or C_i clears itself with high probability.

So far, we have only shown that the lemma holds for the first t_i^j (i.e., t'). For the induction step, we consider an arbitrary t_i^j. By the induction hypothesis, we can assume that all previous such time-slots have already been dealt with. In other words, all previously considered time-slots $t_{i'}^{j'}$ have either lead to a clearance of circle $C_{i'}$ or the sum of probabilities in $C_{i'}$ has decreased below the threshold ηd^2. Immediately after a clearance, the sum of sending probabilities in a circle C_i is at most ηd^2, which is the sending probability in the last round of the algorithm. This is true because only one node in the circle remains undecided. All other nodes will stop sending on channel Γ_1. By Lemma 5.5, the sum of sending probabilities in all neighboring circles (both the cleared and the not cleared ones) is bounded by $3\eta d^2$ in the interval $[t_i^j \ldots t_i^j+\varphi-1]$ (otherwise, this circle would have been considered before t_i^j). Therefore, we know that the bounds (1) and (2) hold with high probability. And consequently, the computation to show the induction step is the same as the one for the base case t' and it also holds that $\overline{P_{term}} \in \mathrm{o}\big(\frac{1}{n^2}\big)$.

Each step of the induction only holds with high probability. But, because there are n nodes to be decided and at most nonempty n circles C_i, the number of induction steps t_i^j is bounded by n. Hence, the probability that the lemma holds for all steps is at least $\big(1-\mathrm{o}\big(\frac{1}{n^2}\big)\big)^n \geq 1-\mathrm{o}\big(\frac{1}{n}\big)$, which concludes the proof. □

Using Lemma 5.6, we can now compute the expected number of dominators in each circle C_i. In the analysis, we will separately compute the number of dominators *before* and *after* the termination (i.e. the first clearance) of C_i. To prove our results, we will need three more lemmas.

Lemma 5.7. *Let C be the number of collisions (more than one node is sending in one time-slot on Γ_1) in a circle C_i. The expected number of collisions in C_i before its termination is $\mathrm{E}[C] < 5$. Further, $C < 6\log n$ with probability at least $1-\mathrm{o}\big(\frac{1}{n^2}\big)$.*

Proof. Only channel Γ_1 is considered in this proof. We assume that C_i is not yet terminated and we define the following events

- A : Exactly one node in D_i is sending
- X : More than one node in C_i is sending
- Y : At least one node in C_i is sending
- Z : Some node in $D_i \setminus C_i$ is sending

For the proof, we consider only rounds in which at least one node in C_i sends. (There will be no new dominators in C_i if no node sends). We want to get a bound for the conditional probability $\mathrm{P}[A \mid Y]$ that exactly one node in D_i is sending and this one node is located in C_i. Using $\mathrm{P}[Y \mid X]=1$ and the fact that Y and Z are independent, we get

$$
\begin{aligned}
\mathrm{P}[A \mid Y] &= \mathrm{P}\left[\overline{X} \mid Y\right]\cdot\mathrm{P}\left[\overline{Z} \mid Y\right] \\
&= \mathrm{P}\left[\overline{X} \mid Y\right]\cdot\mathrm{P}\left[\overline{Z}\right] \\
&= (1-\mathrm{P}[X \mid Y])\cdot(1-\mathrm{P}[Z]) \\
&= \left(1-\frac{\mathrm{P}[X]\cdot\mathrm{P}[Y \mid X]}{\mathrm{P}[Y]}\right)\cdot(1-\mathrm{P}[Z]) \\
&= \left(1-\frac{\mathrm{P}[X]}{\mathrm{P}[Y]}\right)\cdot(1-\mathrm{P}[Z])\,. \qquad (5)
\end{aligned}
$$

We can now compute bounds for the probabilities $\mathrm{P}[X]$, $\mathrm{P}[Y]$, and $\mathrm{P}[Z]$:

$$
\begin{aligned}
\mathrm{P}[X] &= 1-\prod_{k\in C_i}(1-p_k)-\sum_{k\in C_i}\left(p_k\prod_{\substack{l\in C_i\\ l\neq k}}(1-p_l)\right) \\
&\leq 1-\left(\frac{1}{4}\right)^{\sum_{k\in C_i}p_k}-\sum_{k\in C_i}p_k\cdot\left(\frac{1}{4}\right)^{\sum_{k\in C_i}p_k} \\
&= 1-\left(1+\sum_{k\in C_i}p_k\right)\left(\frac{1}{4}\right)^{\sum_{k\in C_i}p_k} \qquad (6) \\
\mathrm{P}[Y] &= 1-\prod_{k\in C_i}(1-p_k) \;\geq\; 1-\left(\frac{1}{e}\right)^{\sum_{k\in C_i}p_k}. \qquad (7)
\end{aligned}
$$

The first inequality for $\mathrm{P}[X]$ follows from Fact 5.1 and inequality (4). The inequality for $\mathrm{P}[Y]$ also follows from Fact 5.1. In the proof for Lemma 5.6, we have already computed a bound for P_0, the probability that no node in $D_i \backslash C_i$ sends. Using this result, we can write $\mathrm{P}[Z]$ as

$$
\mathrm{P}[Z] = 1-\prod_{C_j\in D_i\backslash C_i}\prod_{k\in C_j}(1-p_k) \underset{\text{Eq. (3)}}{\leq} 1-\left(\frac{1}{4}\right)^{\frac{3}{4}}. \qquad (8)
$$

Plugging inequalities (6), (7), and (8) into equation (5) for $\mathrm{P}[A \mid Y]$, it can be shown that the term $\mathrm{P}[X]/\mathrm{P}[Y]$ is maximized for $\sum_{k\in C_i}p_k = 3\eta d^2$ and therefore

$$
\begin{aligned}
\mathrm{P}[A \mid Y] &= \left(1-\frac{\mathrm{P}[X]}{\mathrm{P}[Y]}\right)\cdot(1-\mathrm{P}[Z]) \\
&\geq \left(1-\frac{1-(1+3\eta d^2)\left(\frac{1}{4}\right)^{3\eta d^2}}{1-\left(\frac{1}{e}\right)^{3\eta d^2}}\right)\left(\frac{1}{4}\right)^{\frac{3}{4}}.
\end{aligned}
$$

Since this expression is minimized for $d=1$, we obtain $\mathrm{P}[A \mid Y] \geq 0.211$ by plugging in all values. Hence, whenever a node in C_i sends, C_i terminates with constant probability at least $\mathrm{P}[A \mid Y]$. This allows us to compute the expected number of collisions in C_i before the termination of C_i as a geometric distribution:

$$
\mathrm{E}[C] \;=\; \frac{1}{\mathrm{P}[A \mid Y]} \;\leq\; 5.
$$

The high probability result can be derived as

$$
\mathrm{P}[C \geq 6\log n] \;=\; (1-\mathrm{P}[A \mid Y])^{6\log n} \in \mathrm{O}\big(n^{-2}\big)\,.
$$

□

So far, we have shown that the number of collisions before the clearance of C_i is constant in expectation. The next lemma shows that the number of *new dominators per collision* is also constant. In a collision, each of the sending nodes may already be dominator. Hence, if we assume that every sending node in a collision is a new dominator, we obtain an upper bound for the true number of new dominators.

LEMMA 5.8. *Let D be the number of nodes in C_i sending in a time-slot and let Φ denote the event of a collision. Given the occurrence of a collision, the expected number of sending nodes (i.e., new dominators) is $\mathrm{E}[D \mid \Phi] \in \mathrm{O}(1)$. Furthermore, $\mathrm{P}[D < 3\log n/\log\log n \mid \Phi]$ holds with high probability.*

PROOF. Let m, $m \leq n$, be the number of nodes in C_i and $\mathrm{M} = \{1 \dots m\}$. D is a random variable denoting the number of sending nodes in C_i in a given time-slot. We define $A_k := \mathrm{P}[D = k]$ as the probability that exactly k nodes send. For example, the probability that exactly two nodes in C_i send is

$$A_2 \;=\; \sum_{k \in C_i} \left(p_k \cdot \prod_{\substack{l \in C_i \\ l \neq k}} (1 - p_l) \right).$$

Let $\binom{\mathrm{M}}{k}$ be the set of all k-subsets of M (subsets of M having exactly k elements). We define A'_k as

$$A'_k \;:=\; \sum_{\mathrm{Q} \in \binom{\mathrm{M}}{k}} \prod_{i \in \mathrm{Q}} \frac{p_i}{1 - p_i}.$$

We can then write A_k as

$$\begin{aligned} A_k \;&=\; \sum_{\mathrm{Q} \in \binom{\mathrm{M}}{k}} \left(\prod_{i \in \mathrm{Q}} p_i \cdot \prod_{i \notin \mathrm{Q}} (1 - p_i) \right) \\ &=\; \left(\sum_{\mathrm{Q} \in \binom{\mathrm{M}}{k}} \prod_{i \in \mathrm{Q}} \frac{p_i}{1 - p_i} \right) \cdot \prod_{i=1}^{m} (1 - p_i) \\ &=\; A'_k \cdot \prod_{i=1}^{m} (1 - p_i). \end{aligned} \tag{9}$$

FACT 5.9. *The following recursive inequality holds between two subsequent A'_k:*

$$\begin{aligned} A'_k \;&\leq\; \frac{1}{k} \sum_{i=1}^{m} \frac{p_i}{1 - p_i} \cdot A'_{k-1} \\ A'_0 \;&=\; 1. \end{aligned}$$

PROOF. The base case $A'_0 = 1$ follows directly from equation (9) and the fact that the probability A_0 that no node sense is $\prod_{i=1}^{m} (1 - p_i)$. For general A'_k, we have to group the terms $\prod_{i \in \mathrm{Q}} \frac{p_i}{1 - p_i}$ in such a way that we can factor out A'_{k-1}:

$$\begin{aligned} A'_k \;&=\; \sum_{\mathrm{Q} \in \binom{\mathrm{M}}{k}} \prod_{j \in \mathrm{Q}} \frac{p_j}{1 - p_j} \\ &=\; \frac{1}{k} \sum_{i=1}^{m} \left(\frac{p_i}{1 - p_i} \cdot \sum_{\mathrm{Q} \in \binom{\mathrm{M} \setminus \{i\}}{k-1}} \prod_{j \in \mathrm{Q}} \frac{p_j}{1 - p_j} \right) \\ &\leq\; \frac{1}{k} \sum_{i=1}^{m} \left(\frac{p_i}{1 - p_i} \cdot \sum_{\mathrm{Q} \in \binom{\mathrm{M}}{k-1}} \prod_{j \in \mathrm{Q}} \frac{p_j}{1 - p_j} \right) \\ &=\; \frac{1}{k} \sum_{i=1}^{m} \frac{p_i}{1 - p_i} \cdot \left(\sum_{\mathrm{Q} \in \binom{\mathrm{M}}{k-1}} \prod_{j \in \mathrm{Q}} \frac{p_j}{1 - p_j} \right) \\ &=\; \frac{1}{k} \sum_{i=1}^{m} \frac{p_i}{1 - p_i} \cdot A'_{k-1}. \end{aligned}$$

□

We now continue the proof of Lemma 5.8. The conditional expected value $\mathrm{E}[D \mid \Phi]$ is

$$\mathrm{E}[D \mid \Phi] \;=\; \sum_{i=0}^{m} (i \cdot \mathrm{P}[D = i \mid \Phi]) \;=\; \sum_{i=2}^{m} B_i. \tag{10}$$

where B_i is defined as $i \cdot \mathrm{P}[D = i \mid \Phi]$. For $i \geq 2$, the conditional probability reduces to

$$\mathrm{P}[D = i \mid \Phi] = \frac{\mathrm{P}[D = i]}{\mathrm{P}[\Phi]}. \tag{11}$$

In the next step, we consider the ratio between two consecutive terms of sum (10).

$$\begin{aligned} \frac{B_{k-1}}{B_k} \;&=\; \frac{(k-1) \cdot \mathrm{P}[D = k-1 \mid \Phi]}{k \cdot \mathrm{P}[D = k \mid \Phi]} \\ &\underset{\text{Eq. (11)}}{=}\; \frac{(k-1) \cdot \mathrm{P}[D = k-1]}{k \cdot \mathrm{P}[D = k]} \\ &=\; \frac{(k-1) \cdot A_{k-1}}{k \cdot A_k} = \frac{(k-1) \cdot A'_{k-1}}{k \cdot A'_k}. \end{aligned}$$

It follows from Fact 5.9, that each term B_k can be upper bounded by

$$\begin{aligned} B_k \;&=\; \frac{k A'_k}{(k-1) A'_{k-1}} \cdot B_{k-1} \\ &\underset{\text{Fact 5.9}}{\leq}\; \frac{k \left(\frac{1}{k} \sum_{i=1}^{m} \frac{p_i}{1 - p_i} \cdot A'_{k-1} \right)}{(k-1) A'_{k-1}} \cdot B_{k-1} \\ &=\; \frac{1}{k-1} \sum_{i=1}^{m} \frac{p_i}{1 - p_i} \cdot B_{k-1} \\ &\leq\; \frac{2}{k-1} \sum_{i=1}^{m} p_i \cdot B_{k-1}. \end{aligned}$$

The last inequality follows from $\forall i : p_i < 1/2$ and $p_i \leq 1/2 \Rightarrow \frac{p_i}{1 - p_i} \leq 2p_i$.

From the definition of B_k, it naturally follows that $B_2 \leq 2$. Furthermore, we can bound the sum of sending probabilities $\sum_{i=1}^{m} p_i$ using Lemmas 5.3 and 5.6 to be less than

$3\eta d^2 \cdot \left(\frac{5}{d^2} + \frac{15}{d} + 11\right) \leq \frac{3}{4}$. We can thus sum up over all B_i recursively in order to obtain $\mathrm{E}[D \mid \Phi]$:

$$\begin{aligned}\mathrm{E}[D \mid \Phi] &= \sum_{i=2}^{m} B_i \\ &\leq 2 + \sum_{i=3}^{\infty}\left[\frac{2}{(i-1)!}\left(\frac{3}{4}\right)^{i-2}\right] \\ &= \frac{3}{8}e^{3/4} - \frac{3}{8} \leq 2.98.\end{aligned}$$

In order to derive the high probability result, we solve the recursion of Fact 5.9 and obtain

$$A'_k \leq \frac{1}{k!}\left(\sum_{i=1}^{m}\frac{p_i}{1-p_i}\right)^k. \qquad (12)$$

The probability $P_+ := \mathrm{P}\left[D \geq \frac{3\log n}{\log\log n} \mid \Phi\right]$ is

$$\begin{aligned}P_+ &= \sum_{k=\lceil\frac{3\log n}{\log\log n}\rceil}^{n} A_k \leq \sum_{k=\lceil\frac{3\log n}{\log\log n}\rceil}^{n} A'_k \\ &\underset{\text{Eq. (12)}}{\leq} \sum_{k=\lceil\frac{3\log n}{\log\log n}\rceil}^{n}\left[\frac{1}{k!}\cdot\left(\sum_{i=1}^{m}\frac{p_i}{1-p_i}\right)^k\right] \\ &\leq \sum_{k=\lceil\frac{3\log n}{\log\log n}\rceil}^{n}\left[\frac{1}{k!}\cdot\left(2\cdot\sum_{i=1}^{m}p_i\right)^k\right] \\ &\leq \left(n - \left\lceil\frac{3\log n}{\log\log n}\right\rceil\right)\cdot\frac{\left(2\cdot\sum_{i=1}^{m}p_i\right)^{\lceil\frac{3\log n}{\log\log n}\rceil}}{\lceil\frac{3\log n}{\log\log n}\rceil!} \\ &\leq n\cdot\frac{(3/4)^{\lceil\frac{3\log n}{\log\log n}\rceil}}{\lceil\frac{3\log n}{\log\log n}\rceil!} \in \mathrm{O}(n^{-2}).\end{aligned}$$

The last key lemma shows that the expected number of new dominators *after* the termination of circle C_i is also constant.

LEMMA 5.10. *Let A be the number of new dominators after the termination of C_i. Then, $A \in \mathrm{O}(1)$ with high probability.*

PROOF. We define B and B_i as the set of dominators in D_i and C_i, respectively. Immediately after the termination of C_i, only one node in C_i remains sending on channel Γ_1, because all others will be decided. By Lemmas 5.7 and 5.8, we can bound the number of dominators in a C_i with high probability as $|B_i| \leq \tau' \log^2 n/\log\log n$ for a small constant τ'. Potentially, all $C_j \in D_i$ are already terminated and therefore,

$$1 \leq |B| \leq \left(\frac{5}{d^2} + \frac{15}{d} + 11\right)\cdot\tau'\frac{\log^2 n}{\log\log n} \qquad (13)$$

with high probability. For simplicity, we write $\tau(d) := \tau'\left(\frac{5}{d^2} + \frac{15}{d} + 11\right)$.

We distinguish the two cases $1 \leq |B| \leq \tau(d)\log n/\log\log n$ and $\tau(d)\log n/\log\log n < |B| \leq \tau(d)\log n^2/\log\log n$. We consider channels Γ_2 and Γ_3 in the first and second case, respectively. Particularly, we prove that in the first (second) case, a new node will receive a message on Γ_2 (Γ_3) with high probability during the waiting period at the beginning of the algorithm.

Consider case one, i.e. $1 \leq |B| \leq \tau(d)\log n/\log\log n$. The probability P_0 that one dominator is sending alone on channel Γ_2 is

$$P_0 = |B|\cdot q_1\cdot(1-q_1)^{|B|-1}$$

where $q_1 := \eta d^2\log\log n/\log n$ as defined in the algorithm.

As this is a concave function in $|B|$, it is sufficient to consider the two extreme values. For $|B| = 1$, we get $P_0 = q_1 = \eta d^2\log\log n/\log n$ and for $|B| = \tau(d)\log n/\log\log n$, $n \geq 2$, we have

$$\begin{aligned}P_0 &\geq \tau(d)\eta d^2\cdot\left(1 - \frac{\eta d^2\log\log n}{\log n}\right)^{\frac{\tau(d)\log n}{\log\log n}} \\ &\geq \tau(d)\eta d^2\cdot\left(1 - \frac{d^2\tau(d)/2^\beta}{\frac{\tau(d)\log n}{\log\log n}}\right)^{\frac{\tau(d)\log n}{\log\log n}} \\ &\geq \tau(d)\eta d^2 e^{-\tau(d)\eta d^2}\left(1 - \frac{\left(\tau(d)\eta d^2\right)^2}{\frac{\tau(d)\log n}{\log\log n}}\right) \\ &\geq \tau(d)\eta d^2 e^{-\tau(d)\eta d^2}\left(1 - d^4\eta^2\tau(d)\right) \in \Omega(1).\end{aligned}$$

A newly awakened node in a terminated circle C_i will not send during the first $\alpha\lceil\log^2 n/(d^2\log\log n)\rceil$ time-slots. If during this period, the node receives a message (without collision) from an existing dominator, it will become decided and hence, will not become dominator. The probability P_{no} that such an already covered node does *not* receive any messages from an existing dominator during the first $\alpha\lceil\log^2 n/(d^2\log\log n)\rceil$ time-slots is asymptotically bounded by

$$\begin{aligned}P_{no} &\leq \left(1 - \frac{\eta d^2\log\log n}{\log n}\right)^{\alpha\cdot\lceil\log^2 n/(d^2\log\log n)\rceil} \\ &\underset{\text{Fact 5.2}}{\leq} e^{-\eta\alpha\log n} \in \mathrm{O}(n^{-4}).\end{aligned}$$

This shows that the probability of new dominators emerging in C_i after the termination of C_i is small and with high probability, the number of new dominators is bounded by a constant in this case.

The analysis in the second case follows along the same lines, the only difference being that we consider channel Γ_3 instead of Γ_2. For $|B| = \tau\log n/\log\log n$, we get

$$\begin{aligned}P_0 &\geq \frac{\eta d^2\tau(d)}{\log n}\left(1 - \frac{\eta d^2\log\log n}{\log^2 n}\right)^{\frac{\tau(d)\log n}{\log\log n}} \\ &\geq \frac{\eta d^2\tau(d)}{\log n}\cdot\left(1 - \frac{\eta d^2\tau(d)/(\log n)}{\tau(d)\log n/\log\log n}\right)^{\frac{\tau(d)\log n}{\log\log n}} \\ &\geq \frac{\eta d^2\tau(d)}{\log n}e^{-\frac{\eta d^2\tau(d)}{\log n}}\left(1 - \eta^2 d^4\tau(d)\right) \in \Omega(1/\log n).\end{aligned}$$

For $|B| = \tau(d)\log^2 n/\log\log n$, it can be shown that $P_0 \in \Omega(1)$ and hence, the remainder of the analysis is analogous to the first case. □

We are now ready to prove the following theorem.

THEOREM 5.11. *The expected number of dominators in circle C_i is $\mathrm{E}[D] \in \mathrm{O}(1)$.*

PROOF. We consider a circle C_i. By Lemma 5.7, the expected number of collisions before the termination of C_i is less than 5. Lemma 5.8 states that the expected number of new dominators per collision is not higher than 2.98. Because C and $D \mid \Phi$ are independent events, we can compute the expected number of dominators in C_i before the termination of C_i as

$$\mathrm{E}[D] \;=\; \mathrm{E}[C] \cdot \mathrm{E}[D \mid \Phi] \;\leq\; 15 \;\in\; \mathrm{O}(1)$$

By Lemma 5.10, the number of dominators emerging after the termination of C_i is also constant. $\square$

Note that the exact value of the constant in Theorem 5.11 is an artefact of the analysis and does not closely reflect the algorithm's performance in real settings. In Section 8, we show that, for reasonable parameter values, the real bounds are close to 2.

THEOREM 5.12. *The algorithm computes a correct dominating set in time*

$$\mathrm{O}\left(\frac{\log N}{d^2}\left(\log \Delta + \frac{\log N}{\log \log N}\right)\right)$$

and achieves an approximation ratio of $\mathrm{O}(1/d^2)$ *in expectation.*

PROOF. Correctness and runtime follow directly from Theorems 4.1 and 4.2. For the approximation ratio, Theorem 5.11 bounds the number of dominators in each circle C_i by a constant. On the other hand, even the optimal solution must choose at least one dominator in D_i. The Theorem then follows from the fact that D_i covers at most $5/d^2 + 15/d + 11$ circles C_i by Lemma 5.3. $\square$

6. SINGLE-CHANNEL

In this section, we analyze the single-channel scenario. We prove that the constant approximation-ratio (for d being constant) also holds in this case, by showing how each time-slot in the multi-channel model can be simulated by a number of time-slots in the single-channel model. The time complexity of the algorithm remains polylogarithmic.

Let s and t be time-slots in the single-channel and multi-channel model, respectively. We write $suc(t) = 1$ if a message is successfully transmitted in time-slot t and $suc(t) = 0$ otherwise.

LEMMA 6.1. *Time-slot t can be simulated with* $\mathrm{O}(\log^3 n/d^2)$ *time-slots s_i, $i \in [1 \ldots 3\beta \log^3 n/d^2]$, for a large enough constant β such that $suc(t) = 1 \Leftrightarrow \exists i : suc(s_i) = 1$ with probability* $1 - \mathrm{O}(\frac{1}{n^2})$.

PROOF. We first investigate the critical cases by analyzing the different sending possibilities which can occur in the multi-channel case (channels Γ_1, Γ_2, and Γ_3) and how they map to the single-channel model.

Γ_1	Γ_2	Γ_3	Multi	Single	Critical
0	0	0	0	0	no
0	1	0	1	1	no
0	1	1	1	0	yes
1	1	1	1	0	yes
0	≥ 2	0	0	0	no
1	≥ 2	0	1	0	yes
1	≥ 2	1	1	0	yes
≥ 2	≥ 2	1	1	0	yes
≥ 2	≥ 2	≥ 2	0	0	no

The table shows some of the possible cases. The columns Γ_1, Γ_2, and Γ_3 denote how many senders are sending on these channels in a given time-slot. The next two columns show whether or not the transmission was successful, depending on the number of channels used. For the single-channel case, we assume that all senders sending on any channel are sending on a common channel Γ. The critical cases are those in which a node receives a message in the multi-channel case, but does not receive it in the single-channel case, due to a collision. When simulating three channels by a single channel, we must ensure that a message can be successfully transmitted (without collision) in these critical cases.

We write $send(t) = 1$ if a sender sends in time-slot t and $send(t) = 0$, otherwise. Further, we use the abbreviations $\lambda := \beta \log^3 n/d^2$ and $p := d^2/\log^2 n$. Each node simulates time-slot t by 3λ single-channel time-slots $s_1 \ldots s_{3\lambda}$ in the following way:

$$\begin{aligned}
send(t) = 0 \;&\Rightarrow\; \forall s_i \in [s_1 \ldots s_{3\lambda}] : \\
&\qquad send(s_i) := 0 \\
send(t) = 1 \;&\Rightarrow\; \forall s_i \in [s_1 \ldots s_\lambda \;,\; s_{2\lambda} \ldots s_{3\lambda}] : \\
&\qquad send(s_i) := 0 \\
send(t) = 1 \;&\Rightarrow\; \forall s_i \in [s_\lambda \ldots s_{2\lambda}] : \\
&\qquad send(s_i) := \begin{cases} 1, & \text{with probability } p \\ 0, & \text{with probability } 1-p \end{cases}
\end{aligned}$$

In words, each node which sends on Γ_1, Γ_2, or Γ_3 in a time-slot t sends randomly with probability $d^2/\log^2 n$ in the λ time-slots $[s_\lambda \ldots s_{2\lambda}]$ on channel Γ. We call $[s_\lambda \ldots s_{2\lambda}]$ *sending period*, $[s_1 \ldots s_\lambda]$ and $[s_{2\lambda} \ldots s_{3\lambda}]$ *quiet periods.*

Obviously, if there is more than one sender, they may choose the same or overlapping time-slots, which will lead to collisions. Unless there is at least one time-slot in which exactly one sender is sending, the message is not transmitted successfully. Thus, there is a non-zero probability that sending a message fails in the critical cases, as defined above. We will now show, however, that this probability becomes sufficiently small to make sure the algorithm works the same way as in the multi-channel case.

Let T be the set of sending nodes in time-slot t. Due to asynchronous wake-up, we can not assume that the periods $[s_1 \ldots s_{3\lambda}]$ of sending nodes $v \in T$ are aligned. It is easy to observe, however, that the probability of a successful transmission is minimized when these periods are exactly aligned. If some nodes $v \in T$ are in the sending period while others are in a quiet period, the probability of a successful transmission (exactly one node sends in a time-slot s) is bigger compared to the case when all sending nodes are in the sending period at the same time. Consequently, we only have to consider the case of perfect alignment between sending periods.

By Lemma 5.8, we know that the number of sending nodes on channel Γ_1 in a given time-slot does not exceed $\log n/\log\log n$ with high probability. By Equation (13), we know that the number of nodes sending on Γ_2 and Γ_3 is bounded by $\tau(d)\log^2 n/\log\log n < \tau(d)\log^2 n$ (see remark at the end of the proof).

The probability P_1 that exactly one node $v \in T$ sends in a time-slot s is

$$P_1 = \frac{d^2|T|}{\log^2 n}\left(1 - \frac{d^2}{\log^2 n}\right)^{|T|-1}$$

Since this is a concave function, we again have to consider the cases $|T| = 2$ and $|T| = 2\tau(d)\log^2 n + 1$. In the first case, the probability P_{no} that no message is successfully transmitted in the entire sending period is

$$\begin{aligned} P_{no} &= (1-P_1)^\lambda \\ &\leq \left(1 - \frac{2d^2}{\log^2 n}\left(1 - \frac{d^2}{\log^2 n}\right)\right)^{\frac{\beta \log^3 n}{d^2}} \\ &= \left(1 - \frac{2d^2\log^2 n - 2d^4}{\log^4 n}\right)^{\frac{\beta \log^4 n}{d^2 \log n}} \\ &\underset{\text{Fact 5.2}}{\leq} e^{-2\beta(\log n - \frac{d^2}{\log n})} \in \mathrm{O}\left(\frac{1}{n^{2\beta}}\right). \end{aligned}$$

As for the second case, $|T| = 2\tau(d)\log^2 n + 1$, we have

$$\begin{aligned} P_{no} &= (1-P_1)^\lambda \\ &\leq \left(1 - 2d^2\tau(d)\left(1 - \frac{d^2}{\log^2 n}\right)^{2\tau(d)\log^2 n}\right)^{\frac{\beta \log^3 n}{d^2}} \\ &\leq \left(1 - 2d^2\tau(d)e^{-2d^2\tau(d)}\right)^{\beta \log^3 n} \in \mathrm{O}\left(\frac{1}{n^{\beta \log^2 n}}\right). \end{aligned}$$

Since the same computation holds for all three channels, a message from each channel is successfully transmitted with high probability. □

Remark: The simulation's running time can easily be improved by an additional $\log\log n$ factor by using the improved bound $\tau(d)\log^2 n/\log\log n$ on the maximum number of sending nodes, rather than the weaker $\tau(d)\log^2 n$. We chose to present the slightly worse result for the sake of readability.

COROLLARY 6.2. *The dominator algorithm in the single-channel model has time-complexity* $\mathrm{O}(polylog(n)/d^4)$. *All critical steps are executed like in the multi-channel algorithm with probability at least* $1 - \mathrm{O}\left(\frac{\log^2 n}{d^2 n^{2\beta-1}}\right)$, *for a constant* β.

PROOF. It only remains to show correctness. We compute the probability P that at least one step of the entire algorithm's execution is *not* handled correctly by our single-channel simulation. In the multi-channel case, the algorithm's execution takes no more $\mathrm{O}(1)\cdot n\log^2 n/d^2$ steps. The probability P that none of these critical steps fails is

$$P \geq \left(1 - \frac{1}{n^{2\beta}}\right)^{\mathrm{O}(1)\cdot n\log^2 n/d^2} \in 1 - \mathrm{O}\left(\frac{\log^2 n}{d^2 n^{2\beta-1}}\right).$$

□

Thus, we have shown that even with a single communication channel, we can compute an asymptotically optimal clustering in polylogarithmic time. As for a constant approximation, algorithms with sublogarithmic running time appear to be impossible, future improvements may focus on reducing the logarithm's power.

7. APPLICATIONS

One way to look at our algorithm is to regard it as a first step towards analyzing (and hopefully solving) the characteristic problems existing during and immediately after the deployment of ad hoc and sensor networks. From this point of view, the algorithm serves as an example that our initialization model allows to obtain interesting results. We believe, however, that the algorithm in its current state has relevance beyond being a mere building block for future work or serving as an example for a proposed model. In this section, we briefly discuss the design of a straight-forward protocol for energy-saving which is based on our algorithm.

As mentioned in the introduction, a problem frequently encountered in sensor networks is that a significant fraction of the node's energy supply is used up during the initialization phase. Consider for instance a scenario where sensor nodes are installed indoors and the installation process requires a few days time. When being deployed (installed), nodes become switched on one after the other. Before neighboring nodes have agreed on a common sleep/listen schedule, they must basically remain awake all the time in order to learn about new neighbors. Considering the scarcity of battery power in sensor networks, remaining switched on for a period of days is costly.

The protocol consists of a periodic sleep and listen schedule, a period being of length $\pi_p := \pi_s + \pi_l$. Each period consists of a *sleeping phase* of length π_s, followed by a typically much shorter *listen phase* of length π_l. Upon waking up, nodes are constantly awake and execute Algorithm 1. Instead of sending dummy messages, however, dominators will use their messages to "synchronize" associated nodes, such that all nodes in the same cluster listen and sleep at the same time.

More precisely, the protocol works as follows. Each node v maintains a local[2] counter c_v, initially set to 0 for all nodes. Dominator nodes increment their counter by 1 in each time-slot and attach the current counter value to each message they send. Once the counter reaches π_p, dominators reset their counter, i.e., $c_v := 0$. Intuitively, the counter value c_v denotes how many time-slots have passed since the start of the last sleep phase in v's schedule.

A non-dominator node u associates itself with the first dominator v from which it hears a message, adopting v's sleep/listen schedule. Particularly, upon receiving its first message from a dominator (containing counter c_v), a node u immediately goes to sleep mode for exactly $\min\{\pi_s - c_v, 0\}$ time-slots.

Observe that this simple procedure ensures that all non-dominator nodes in the same cluster (having the same dominator) wake up and sleep at the same time. This, in turn, allows non-dominators to go to sleep mode quickly, i.e., after listening for no more than $\mathrm{O}(\log^2 N)$ time-slots. Clearly, dominator nodes require more energy since they frequently have to send on channels Γ_2 and Γ_3. Each time a dominator has sent a message, it can randomly choose its next sending time-slot. By going into sleep mode between these two subsequent sending time-slots, even dominators can reduce their energy-consumption during the initialization phase. More precisely, a dominator sends on channels Γ_2 and Γ_3 with probabilities $\eta d^2 \log\log N/\log N$ and $\eta d^2 \log\log N/\log^2 N$, respectively. Hence, dominators need only be awake for roughly a fraction of $\log\log N/\log N$ of all time-slots during the initialization.

[2]recall that due to asynchronous wake-up, there are no synchronized or global clocks.

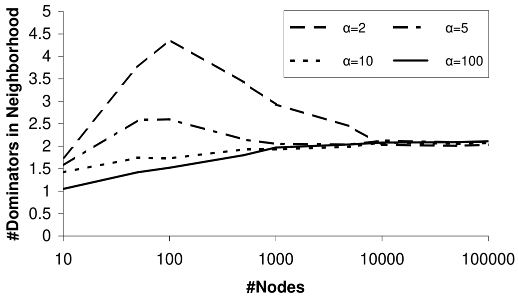

Figure 3: Synchronous Wake-up (p=1)

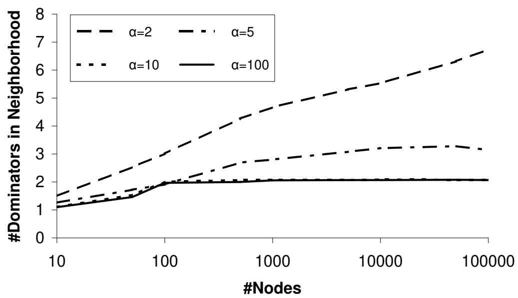

Figure 4: Asynchronous Wake-up (p=10^{-5})

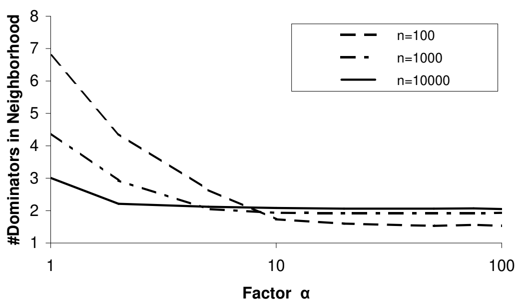

Figure 5: Synchronous Wake-up (p=1)

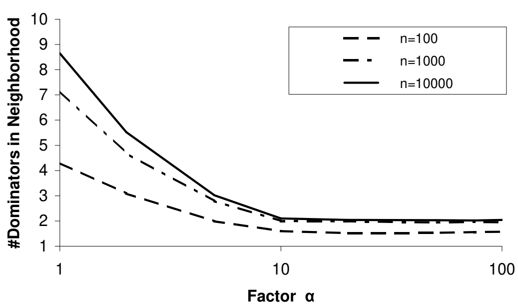

Figure 6: Asynchronous Wake-up (p=10^{-5})

8. SIMULATION

In this section, we give an indication of the efficiency of the algorithm in a variety of test settings. We show how the different parameters influence the quality of the dominating set, as well as the runtime of individual nodes. We have implemented a test setting in which n nodes are randomly distributed in a 5×5 square. Each node's transmission range

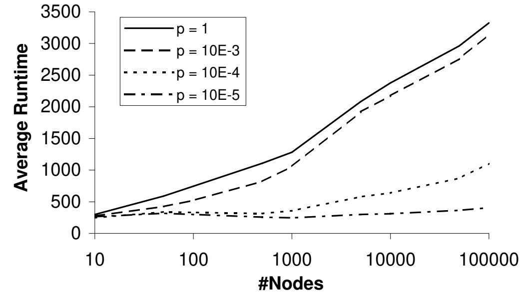

Figure 7: Runtime ($\alpha = 10$)

is 1. Simulating the behavior of a quasi unit disk graph is difficult, since it is not clear how to treat "edges" if their existence is unspecified. Therefore, in order to obtain consistent results, we have adopted the unit disk graph model for the simulations, i.e., $d = 1$. Further, $\eta = 2^{-6}$.

The following parameters are used. n is the number of nodes in the network. α corresponds to the parameter as defined in Algorithm 1. Clearly, the larger α, the larger the running time of the algorithm. On the other hand, choosing α too small increases the chance of having multiple dominators in the same vicinity. The third parameter, p, describes the nodes' wake-up behavior. Let s be the number of sleeping nodes at time t. In time-slot $t+1$, each sleeping node wakes up with probability $\frac{np}{s}$. This yields an even distribution in the first p^{-1} time-slots. In the case $p = 1$, all nodes wake up immediately and we have a synchronous wake-up behavior. For small p, the nodes wake up widely dispersed in time. For all our simulations, we assume $N = \Delta = n$. Clearly, for $N > n$, the number of dominators subsides and the runtime increases.

Figures 3 and 4 relate the number of dominators to the number of nodes in the graph. In both cases, the y-axis denotes the average number of dominators in the neighborhood of a node. In Figure 3, all nodes wake up synchronously in the very first time-slot, while in Figure 4, they wake up asynchronously. In both the synchronous and asynchronous case, the number of dominators converges to a constant of about 2, if the parameter α is chosen large enough. Note that this indicates a dominating set of excellent quality. For $\alpha \geq 10$, the differences become negligibly small, even when compared to the value used in the analysis section, where α must be large to obtain all necessary high probability results. If α is chosen too small, however, the number of dominators in each node's neighborhood may increase, particularly in the case of asynchronous wake-up. This increase is caused by a growing number of collisions as well as the fact that the listen-only phase at the beginning of the algorithm becomes too short.

Note that the larger α, the slower the sending probabilities reach large values leading to an increase in the running-time. Thus, it is desirable to choose α as small as possible. The question is, how small can we choose α before the quality of the resulting dominating set starts worsening. We study this issue in Figures 5 and 6. The charts relate the average number of dominators in the neighborhood of a node to the parameter α. It can be seen that the number of dominators reaches its low at around $\alpha = 10$ in the asynchronous case (and already at $\alpha = 5$ in the synchronous case) and remains constant thereafter. For practical purposes, it is therefore sufficient to run the algorithm using $\alpha := 10$ which yields a good running time, as shown in Figure 7.

The y-axis of Figure 7 denotes the average number of time-slots passing between the wake-up of a node and its becoming *decided*. Approximately, for large p, the runtime increases asymptotically as $\mathrm{O}(\log^2 n)$. For small values of p (i.e., for asynchronous wake-up), the average running time is much lower and does not significantly increase in the range $n \in [10 \ldots 100000]$. This chasm is natural considering that each node listens during the first $\alpha\lceil \log^2 n/(d^2 \log\log n)\rceil$ time-slots without sending. In the case of synchronous wake-up, all nodes execute these time-slots simultaneously and they all end up waiting at the same time, causing an increase in the average running time. In the asynchronous case, most nodes wake up at a time when existing dominators are already sending. Most of these nodes will decide not to become a dominator within the first $\alpha\lceil \log^2 n/(d^2 \log\log n)\rceil$ time slots as shown in Lemma 5.10. The running time gap between the synchronous and asynchronous case is therefore a direct consequence of the way the algorithm works.

Concluding the simulation section, we can state that for practical purposes, the parameter α can be chosen to be smaller than in the analysis section. Various simulations indicate that it is sufficient to set $\alpha \approx 10$. This improves the running time while maintaining an excellent quality of the dominating set. The average running time asymptotically increases in $\mathrm{O}(\log^2 n)$ if wake-up is more-or-less synchronous and becomes significantly smaller the more the nodes' wake-up is dispersed in time.

9. RELATED WORK

The problem of finding a minimum dominating set has been proven to be NP-hard [12, 19]. Furthermore, it has been shown in [10] that the best possible approximation ratio for this problem is $\ln \Delta$, Δ being the highest degree in the graph, unless NP has deterministic $n^{\mathrm{O}(\log\log n)}$-time algorithms. For unit disk graphs, the problem remains NP-hard but allows a polynomial time approximation scheme [16].

A multitude of distributed dominating set algorithms have been proposed, both for general graphs [17, 23, 25, 35] and the unit disk graph [2, 11, 32]. In [23], a distributed linear program is used to compute a dominating set of expected size $\mathrm{O}(k\Delta^{2/k} \log \Delta |DS_{\mathrm{OPT}}|)$ in a constant number of rounds with Δ being the maximum degree in an arbitrary graph, and k^2 being the number of communication rounds. On the other hand, it was proven in [22] that in k communication rounds, no algorithm can find a better approximation than the maximum of $\Omega(n^{c/k^2}/k)$ and $\Omega(\Delta^{c/k}/k)$.

The case of unit disk graphs has also attracted much attention. The algorithm proposed in [11] achieves constant approximation in time $\mathrm{O}(\log\log n)$, but relies on the ability of nodes to sense the *distance* to neighboring nodes. The algorithms in [2, 32] make use of the fact that a maximal independent set (MIS) is a constant approximation to the minimum dominating set. Assuming that each node knows the IDs of all its neighbors and that collision-free point-to-point connections between two neighbors are established, the distributed computation of a MIS is relatively straight-forward. If MAC layer issues are considered, however, computing a MIS suddenly becomes a difficult task in view of the hidden-terminal problem. All the above algorithms assume existing point-to-point connections between neighboring nodes.

A first step towards a theoretical analysis of structuring unstructured multi-hop radio networks was made in [21]. In particular, a dominating set based clustering is computed in a model featuring asynchronous wake-up and absence of collision detection. In contrast to this paper, the algorithm in [21] is designed for the unit disk graph model and is not applicable in the single-channel case.

Models related to the one used in this paper have been studied in the context of analyzing the complexity of broadcasting in multi-hop radio network yielding a vast and rich literature, e.g. [4, 9]. In single-hop radio networks, this "broadcast" model has also been the focus of research on two problems called *initialization problem* and *leader election problem* in single-hop radio networks, e.g. [26]. A striking difference to our model is that these algorithms consider *synchronous wake-up*, i.e. nodes have access to a global clock and it is assumed that all nodes start the distributed algorithm at the same time. A model featuring asynchronous wake-up has been studied in recent papers on the so-called *wake-up problem* in single-hop networks [13, 18]. In comparison to our model, these papers define a much *weaker notion of asynchrony*. Particularly, it is assumed that sleeping nodes are woken up by a successfully transmitted message. In a single-hop network, the problem of waking up all nodes reduces to analyzing the number of time-slots until one single message is successfully transmitted without collision. While this definition of asynchrony leads to theoretically interesting problems and algorithms, it does not closely reflect reality.

10. CONCLUSIONS

How can we efficiently structure a wireless ad hoc or sensor network after its "big bang"? In this paper, we have taken a step towards an answer to this question by analyzing the initialization of multi-hop radio networks, that is, the transition from an unstructured to a structured network. Based on the *quasi unit disk graph*, we formulated a model containing many of the harsh, but realistic characteristics of unstructured networks, including the *hidden terminal problem*, *asynchronous wake-up* and *no reliable collision detection*. Further, instead of resorting to the assumption of random node distribution, we give theoretical bounds holding even in the *worst-case*.

The problem of bringing structure into a network (i.e., organizing an efficient medium access scheme) is related to the problem of clustering in absence of an established MAC layer. Explicitly addressing the "chicken-and-egg" problem of the initialization, our randomized algorithm computes an asymptotically optimal dominating set in polylogarithmic time without relying on any MAC layer support. As a straight-forward application of the algorithm, we have described an energy-efficient way of establishing synchronized sleep and listen schedules between nodes in the same cluster.

Aspiring towards the goal of modeling reality as closely as possible, it would be desirable to restrict our model assumptions even further; particularly, we would like to drop the assumption that nodes know an upper bound for the total number of nodes. Unfortunately, the theoretical lower bound proven in [18] thwarts these hopes. Incorporating aspects such as mobility and node-failures into our model and algorithm promises to be an interesting direction for future research.

11. REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless Sensor Networks: A Survey. *Computer Networks Journal*, 38(4):393–422, 2002.

[2] K. Alzoubi, P.-J. Wan, and O. Frieder. Message-Optimal Connected Dominating Sets in Mobile Ad Hoc Networks. In *Proc. of the 3^{rd} ACM Int. Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC)*, pages 157–164, EPFL Lausanne, Switzerland, 2002.

[3] D. J. Baker and A. Ephremides. The Architectural Organization of a Mobile Radio Network via a Distributed Algorithm. *IEEE Transactions on Communications*, COM-29(11):1694–1701, 1981.

[4] R. Bar-Yehuda, O. Goldreich, and A. Itai. On the Time-Complexity of Broadcast in Radio Networks: an Exponential Gap between Determinism and Randomization. In *Proc. 6^{th} ACM Symp. on Principles of Distributed Computing (PODC)*, pages 98–108, 1987.

[5] L. Barrière, P. Fraigniaud, and L. Narayanan. Robust Position-Based Routing in Wireless Ad Hoc Networks with Unstable Transmission Ranges. In *Proc. 5^{th} Int. workshop on Discrete algorithms and methods for mobile computing and communications*, pages 19–27, 2001.

[6] S. Basagni. Distributed Clustering for Ad Hoc Networks. In *Proceedings of the IEEE International Symposium on Parallel Architectures, Algorithms, and Networks (I-SPAN)*, pages 310–315, 1999.

[7] M. Chatterjee, S. K. Das, and D. Turgut. An On-Demand Weighted Clustering Algorithm (WCA) for Ad-Hoc Networks. In *Proceedings of IEEE GLOBECOM 2000*, pages 1697–1701. ACM Press, 2000.

[8] J. Deng and Z. Haas. Dual Busy Tone Multiple Access (DBTMA): A New Medium Access Control for Packet Radio Networks. In *Proceedings of IEEE ICUPC*, volume 1, pages 973–977, 1998.

[9] Y. M. E. Kushilevitz. An $\Omega(D \log(N/D))$ Lower Bound for Broadcast in Radio Networks. *SIAM Journal on Computing*, 27:702–712, 1998.

[10] U. Feige. A Threshold of ln n for Approximating Set Cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

[11] J. Gao, L. Guibas, J. Hershberger, L. Zhang, and A. Zhu. Discrete Mobile Centers. In *Proc. 17^{th} Symposium on Computational Geometry (SCG)*, pages 188–196, 2001.

[12] M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[13] L. Gasieniec, A. Pelc, and D. Peleg. The wakeup problem in synchronous broadcast systems (extended abstract). In *Proceedings of the 19^{th} ACM symposium on Principles of Distributed Computing (PODC)*, pages 113–121, 2000.

[14] M. Gerla and J. Tsai. Multicluster, mobile, multimedia radio network. *ACM/Baltzer Journal of Wireless Networks*, 1(3):255–265, 1995.

[15] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-Efficient Communication Protocol for Wireless Microsensor Networks. In *Proceedings of the 33^{rd} Annual Hawaii International Conference on System Sciences*, pages 3005–3014, 2000.

[16] H. B. Hunt, III, M. V. Marathe, V. Radhakrishnan, S. S. Ravi, D. J. Rosenkrantz, and R. E. Stearns. NC-Approximation Schemes for NP- and PSPACE-hard Problems for Geometric Graphs. *Journal of Algorithms*, 26(2):238–274, 1998.

[17] L. Jia, R. Rajaraman, and R. Suel. An Efficient Distributed Algorithm for Constructing Small Dominating Sets. In *Proc. of the 20^{th} ACM Symposium on Principles of Distributed Computing (PODC)*, pages 33–42, 2001.

[18] T. Jurdzinski and G. Stachowiak. Probabilistic Algorithms for the Wakeup Problem in Single-Hop Radio Networks. In *Proceedings of 13^{th} Annual International Symposium on Algorithms and Computation (ISAAC)*, volume 2518 of *Lecture Notes in Computer Science*, pages 535–549, 2002.

[19] R. M. Karp. Reducibility Among Combinatorial Problems. In *Proc. of a Symposium on the Complexity of Computer Computations*, pages 85–103, 1972.

[20] R. Kershner. The number of circles covering a set. *American Journal of Mathematics*, 62, 1939.

[21] F. Kuhn, T. Moscibroda, and R. Wattenhofer. Radio Network Clustering from Scratch. In *Proceedings of the 12^{th} Annual European Symposium on Algorithms (ESA)*, 2004.

[22] F. Kuhn, T. Moscibroda, and R. Wattenhofer. What Cannot be Computed Locally! In *Proceedings of the 23^{rd} ACM Symposium on the Principles of Distributed Computing (PODC)*, 2004.

[23] F. Kuhn and R. Wattenhofer. Constant-Time Distributed Dominating Set Approximation. In *Proceedings of 22^{nd} ACM Int. Symposium on the Principles of Distributed Computing (PODC)*, pages 25–32, 2003.

[24] F. Kuhn, R. Wattenhofer, and A. Zollinger. Ad-hoc Networks Beyond Unit Disk Graphs. In *Proceedings of the 2003 Joint Workshop on Foundations of Mobile Computing (DIALM-POMC)*, pages 69–78. ACM Press, 2003.

[25] S. Kutten and D. Peleg. Fast Distributed Construction of Small k-Dominating Sets and Applications. *Journal of Algorithms*, 28:40–66, 1998.

[26] K. Nakano and S. Olariu. Energy-Efficient Initialization Protocols for Single-Hop Radio Networks with no Collision Detection. *IEEE Transactions on Parallel and Distributed Systems*, 11(8), 2000.

[27] V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves. Energy-efficient collision-free medium access control for wireless sensor networks. In *Proceedings of the 1^{st} International Conference on Embedded Networked Sensor Systems (SenSys)*, pages 181–192. ACM Press, 2003.

[28] L. G. Roberts. Aloha Packet System with and without Slots and Capture. *ACM SIGCOMM, Computer Communication Review*, 5(2):28–42, 1975.

[29] J. Sharony. An architecture for mobile radio networks with dynamically changing topology using virtual subnets. *Mobile Networks and Applications*, 1(1):75–86, 1996.

[30] P. Sinha, R. Sivakumar, and V. Bharghavan. Enhancing Ad Hoc Routing with Dynamic Virtual Infrastructures. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 1763–1772, 2001.

[31] F. A. Tobagi and L. Kleinrock. Packet Switching in Radio Channels: Part II - The Hidden Terminal Problem in Carrier Sense Multiple Access and the Busy Tone Solution. COM-23(12):1417–1433, 1975.

[32] P. Wan, K. Alzoubi, and O. Frieder. Distributed construction of connected dominating set in wireless ad hoc networks. In *Proceedings of INFOCOM*, 2002.

[33] Y. Wang and X.-Y. Li. Geometric Spanners for Wireless Ad Hoc Networks. In *Proc. of the 22^{nd} Int. Conf. on Distributed Computing Systems (ICDCS)*, 2002.

[34] A. Woo and D.-E. Culler. A transmission control scheme for media access in sensor networks. In *Proc. 7^{th} Int. Conf. on Mobile Computing and Networking (MOBICOM)*, pages 221–235. ACM Press, 2001.

[35] J. Wu and H. Li. On Calculating Connected Dominating Set for Efficient Routing in Ad Hoc Wireless Networks. In *Proc. of the 3^{rd} Int. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM)*, pages 7–14, 1999.

[36] W. Ye, J. Heidemann, and D. Estrin. An Energy-Efficient MAC protocol for Wireless Sensor Networks. In *Proceedings of IEEE INFOCOM*, pages 1567–1576, New York, NY, USA, June 2002.