A Local Broadcast Layer for the SINR Network Model

Magnús M. Halldórsson* mmh@ru.is Reykjavik University

Stephan Holzer† holzer@csail.mit.edu MIT

Nancy Lynch† lynch@csail.mit.edu MIT

Abstract

We present the first algorithm that implements an *abstract MAC (absMAC)* layer in the *Signal-to-Interference-plus-Noise-Ratio (SINR)* wireless network model. We first prove that efficient SINR implementations are not possible for the standard absMAC specification. We modify that specification to an "approximate" version that better suits the SINR model. We give an efficient algorithm to implement the modified specification, and use it to derive efficient algorithms for higher-level problems of global broadcast and consensus.

In particular, we show that the absMAC *progress* property has no efficient implementation in terms of the SINR strong connectivity graph $G_{1-\varepsilon}$, which contains edges between nodes of distance at most $(1-\varepsilon)$ times the transmission range, where $\varepsilon > 0$ is a small constant that can be chosen by the user. This progress property bounds the time until a node is guaranteed to receive *some* message when at least one of its neighbors is transmitting. To overcome this limitation, we introduce the slightly weaker notion of *approximate progress* into the absMAC specification. We provide a fast implementation of the modified specification, based on decomposing the algorithm of [14] into local and global parts. We analyze our algorithm in terms of local parameters such as node degrees, rather than global parameters such as the overall number of nodes. A key contribution is our demonstration that such a local analysis is possible even in the presence of global interference.

Our absMAC algorithm leads to several new, efficient algorithms for solving higher-level problems in the SINR model. Namely, by combining our algorithm with high-level algorithms from [37], we obtain an improved (compared to [14]) algorithm for global single-message broadcast in the SINR model, and the first efficient algorithm for multi-message broadcast in that model. We also derive the first efficient algorithm for network-wide consensus, using a result of [44]. This work demonstrates that one can develop efficient algorithms for solving high-level problems in the SINR model, using graph-based algorithms over a local broadcast abstraction layer that hides the technicalities of the SINR platform such as global interference. Our algorithms do not require bounds on the network size, nor the ability to measure signal strength, nor carrier sensing, nor synchronous wakeup.

*Supported by Icelandic Research Fund grants 120032011 and 152679-051.

†Supported by the following grants: AFOSR Contract Number FA9550-13-1-0042, NSF Award 0939370-CCF, NSF Award CCF-1217506, NSF Award CCF-AF-0937274.

1 Introduction

Two active areas in Distributed Computing Theory are the attempts to understand wireless network algorithms in the *Signal-to-Interference-plus-Noise-Ratio (SINR) model* and *abstract Medium Access Control layers (absMAC)*.

- The SINR model captures wireless networks in a more precise way than traditional graph-based models, taking into account the fact that signal strength decays according to geometric rules and interference and does not simply stop at a certain border.
- Abstract MAC layers (a.k.a Local Broadcast Layers), express guarantees for local broadcast while hiding the complexities of managing message contention. These guarantees include message delivery latency bounds: an *acknowledgment bound* on the time for a sender's message to be received by all neighbors, and a *progress bound* on the time for a receiver to receive some message when at least one neighbor is sending.

In this paper we combine the strengths of both models by abstracting and modularizing broadcast with respect to global interference and decay via the SINR formula. This marks the start of a systematic study that simplifies the development of algorithms for the SINR model. At the same time we provide an example that modularizing and abstracting broadcast using MAC layers is beneficial and does not necessarily result in worse time-bounds than those of the broadcast algorithm being decomposed.

Traditionally, SINR platforms are quite complicated (compared to graph-based platforms), and consequently are very difficult to use directly for designing and analyzing algorithms for higher-level problems[1]. We show how absMACs can help to mask their complexity and make algorithms easier to design. This demonstrates the potential power of absMACs with respect to algorithm design for the SINR model. During this process we point out and overcome inherent difficulties that at first glance seem to separate the MAC layers from the SINR model and other physical models. These difficulties arise because absMACs are graph-based interference models, while physical models capture (global) interference by specific signal-propagation formulas. Overcoming this mismatch is a key difficulty addressed in this work.

We tackle this mismatch by introducing the concept of *approximate progress* into the absMAC specification and analysis. The definition of approximate progress enables us to obtain a good implementation of an absMAC, which enables anyone to immediately transform generic algorithms designed for an absMAC into algorithms for the SINR model. The main observation that inspired the definition of approximate progress is a proof, that no SINR absMAC implementation is able to guarantee fast progress in an SINR-induced graph G, while fast progress can be guaranteed with respect to an approximation $\tilde{G}$ of G. Roughly speaking, as SINR-induced strong connectivity graphs are defined based on discs representing transmission ranges, we choose $\tilde{G} := G_{1-2\varepsilon}$ to approximate $G := G_{1-\varepsilon}$ by making the disc a tiny bit smaller than in G.

This abstraction makes it easier to design algorithms for higher-level problems in the SINR model and has further benefits. One of the most intriguing properties of abstract MAC

[1]We refer by *higher-level problems* to e.g. network-wide broadcast, consensus, or computing fast relaying-routes, max-flow and other problems whose solution requires a good understanding of lower-level problems. Here, we refer by *lower-level problems* to e.g. achieving connectivity, minimizing schedules and capacity maximization, which are better understood by now.

layers is their separation of global from local computation. This is beneficial in two ways. On the one hand this separation allows us to expose useful SINR techniques in the simple setting of local broadcast. On the other hand this separation provides the basic structure to perform an analysis based on local parameters, such as the number of nodes in transmission/communication range and the distance-ratios between them, which is beneficial as pointed out in Section 2.2. Due to this, and the plug-and-play nature of the absMAC theory, we obtain a faster algorithms for global single-message broadcast than [14] and fast algorithms for global multi-message broadcast and consensus in the SINR model. To achieve these results, we simply plug our absMAC implementation and bounds into the results of [37] and [44].

Future Benefits of Abstract MAC Layers in the SINR Model. Many higher-level problems such as global broadcast, routing and reaching consensus are not yet well understood in the SINR model and recently gained more attention [14, 17, 30, 31, 33, 49, 50]. Many of these problems in the algorithmic SINR can be attacked in a structured way by using and implementing absMACs that hide all complications arising from the SINR model and global interference. Using MAC layers, graph-based algorithms can be analyzed in the SINR model even without knowledge of the SINR model and might still lead to almost optimal algorithms as we demonstrate here.

2 Contributions and Related Work

We devote large parts of this article to prove theorems on implementing an absMAC in the SINR model and how to modify the absMAC specification to get better results. Based on these theorems we derive results on higher-level problems in the SINR model. Table 1 summarizes our algorithmic contributions. In the following $G_{1-\varepsilon}$ and $G_{1-2\varepsilon}$ denote two versions of strong connectivity SINR-induced graphs. By $\Delta_{G_{1-\varepsilon}}$ and $\Delta_{G_{1-2\varepsilon}}$ we denote their degree and by $D_{G_{1-\varepsilon}}$ and $D_{G_{1-2\varepsilon}}$ their diameter. The network size is denoted by n and the ratio of the minimum distance to the smallest distance between nodes connected by an edge in $G_{1-\varepsilon}$ is denoted by Λ. Parameter α denotes the path-loss exponent of the SINR model. We state more detailed definitions in Section 4.

Efficient implementation of acknowledgments. Theorem 5.1 transfers Algorithm 1 of [29] and its analysis to implement fast acknowledgments of the absMAC and modifies it to use local parameters. The resulting algorithm performs acknowledgments with probability at least $1-\varepsilon_{ack}$ in time $\mathcal{O}\left(\Delta_{G_{1-\varepsilon}} \log\left(\frac{\Lambda}{\varepsilon_{ack}}\right) + \log(\Lambda)\log\left(\frac{\Lambda}{\varepsilon_{ack}}\right)\right)$. Remark 5.3 provides a close lower bound.

Proof of impossibility of efficient progress. Theorem 6.1 shows that one cannot expect an efficient implementation of progress using the standard definition of absMAC. In particular one cannot implement an absMAC in the SINR model that achieves progress in time $\Delta_{G_{1-\varepsilon}}$ or less. This is not much better than our bound on acknowledgments and therefore inefficient. This lower bound is even true when an optimal schedule for transmissions in the network is computed by a central entity that has full knowledge of all node positions and can choose arbitrary transmission powers for each node. In contrast, all algorithms presented here are fully distributed, use uniform transmission power and do not know the positions of nodes.

The notion of approximate progress. Achieving progress faster than acknowledgment is key to several algorithms designed for absMACs. Motivated by the above lower bound, we relax the notion of progress in the specification of an absMAC to *approximate progress*. Definition 7.1 introduces approximate progress with respect to an approximation (or some subgraph) of the graph in which local broadcast is performed. Although this new notion of approximate progress is weaker than the usual (single-graph) notion of progress, bounds on approximate progress turn out to be strong enough to yield, e.g., good bounds for global broadcast as long as G is, e.g., connected—see Theorem 12.7. The introduction of approximate progress is the main conceptual contribution of this article.

Efficient implementation of approximate progress. We propose an algorithm that implements approximate progress in time $\mathcal{O}\left(\left(\log^{\alpha}(\Lambda) + \log^*\left(\frac{1}{\varepsilon_{approg}}\right)\right)\log(\Lambda)\log\left(\frac{1}{\varepsilon_{approg}}\right)\right)$ with probability at least $1-\varepsilon_{approg}$, see Theorem 9.1. This algorithm is a modification of the global single-message broadcast algorithm of [14] to guarantee approximate progress in a local multi-message environment. This also makes this algorithm suitable for a localized analysis, which enables us to bound on approximate progress depending only on local parameters and the desired success probability. A key issue is that transmissions made below the MAC layer to implement its broadcast service might be highly unsuccessful due to being performed randomly and being prone to interference. Although the absMAC implementation is guaranteed to perform approximate progress with arbitrarily high probability guarantee $1-\varepsilon_{approg}$ (specified by by the user), it is crucial to use very low probability guarantees below the MAC layer. Fast approximate progress for large values of ε_{approg} can only be achieved when this is reflected in probability guarantees below the MAC layer (e.g. by avoiding network wide union bounds, as these require w.h.p. guarantees). This is an important step towards the improved bounds on global broadcast stated below. To argue that despite constant success probability of transmissions during our constructions we can still achieve the desired probability guarantee $1-\varepsilon_{approg}$ for correctness of approximate progress, we 1) argue that global interference from nodes that erroneously participate in the protocol due to previously unsuccessful transmissions does not affect local broadcast much, and 2) bound the local effects of previously unsuccessful transmissions by studying the probability of correct execution of the algorithm in a receiver's neighborhood. This analysis is arguably the main technical contribution of this article.

Global consensus, single-message and multi-message broadcast in the SINR model. We immediately derive an algorithm for global consensus (CONS) in Corollary 5.5 by combining our acknowledgment-bound with a result of [44]. CONS can be achieved with probability at least $1-\varepsilon_{CONS}$ in time $\mathcal{O}\left(D_{G_{1-\varepsilon}}(\Delta_{G_{1-\varepsilon}} + \log(\Lambda))\log\left(\frac{n\Lambda}{\varepsilon_{CONS}}\right)\right)$. Section 12 combines our absMAC implementation with results of [37] in a straightforward way to derive algorithms for global single-message broadcast (SMB) and global multi-message broadcast (MMB). Global SMB can be performed in time $\mathcal{O}\left(\left(D_{G_{1-2\varepsilon}} + \log\left(\frac{n}{\varepsilon_{SMB}}\right)\right)\log^{\alpha+1}(\Lambda)\right)$ with probability at least $1-\varepsilon_{MMB}$. Global MMB can be performed with probability at least $1-\varepsilon_{MMB}$ in time $\mathcal{O}\left(D_{G_{1-2\varepsilon}}\log^{\alpha+1}(\Lambda) + k\left(\Delta_{G_{1-\varepsilon}} + \operatorname{polylog}\left(\frac{nk\Lambda}{\varepsilon_{MMB}}\right)\right)\log\left(\frac{nk}{\varepsilon_{MMB}}\right)\right)$.

Remark 2.1. *All assumptions that we make in the SINR model and in absMACs are listed in Section 4.6 and are mainly adapted from [14] and [32]. Our SINR-related assumptions are*

Task/Bound	Lower bound	Upper bound presented here
f_{ack}	$\Delta_{G_{1-\varepsilon}}$ $^{(+)}$	$\mathcal{O}\left(\Delta_{G_{1-\varepsilon}} \cdot \log\left(\frac{\Lambda}{\varepsilon_{ack}}\right) + \log(\Lambda)\log\left(\frac{\Lambda}{\varepsilon_{ack}}\right)\right)$
f_{prog}	$\Delta_{G_{1-\varepsilon}}^{(*)}$	$\mathcal{O}\left(\Delta_{G_{1-\varepsilon}} \cdot \log\left(\frac{\Lambda}{\varepsilon_{ack}}\right) + \log(\Lambda)\log\left(\frac{\Lambda}{\varepsilon_{ack}}\right)\right)$
f_{approg}	–	$\mathcal{O}\left(\left(\log^{\alpha}(\Lambda) + \log^{*}\left(\frac{1}{\varepsilon_{approg}}\right)\right)\log(\Lambda)\log\left(\frac{1}{\varepsilon_{approg}}\right)\right)$
global SMB	$\Omega\Big(D_{G_{1-\varepsilon}} \log\left(\frac{n}{D_{G_{1-\varepsilon}}}\right) + \log^2(n)\Big)^{(\ddagger)}$	$\mathcal{O}\left(\left(D_{G_{1-2\varepsilon}} + \log\left(\frac{n}{\varepsilon_{SMB}}\right)\right)\log^{\alpha+1}(\Lambda)\right)^{(\dagger)}$
global MMB	$\Omega\Big(D_{G_{1-\varepsilon}} \log\left(\frac{n}{D_{G_{1-\varepsilon}}}\right) + k\log(n) + \log^2(n)\Big)^{(\ddagger)}$	$\mathcal{O}\Big(D_{G_{1-2\varepsilon}} \log^{\alpha+1}(\Lambda) + k\Delta_{G_{1-\varepsilon}} \log\left(\frac{nk}{\varepsilon_{MMB}}\right) + \text{polylog}\left(\frac{nk\Lambda}{\varepsilon_{MMB}}\right)\Big)^{(\dagger)}$
global CONS	–	$\mathcal{O}\left(D_{G_{1-\varepsilon}}(\Delta_{G_{1-\varepsilon}} + \log(\Lambda))\log\left(\frac{n\Lambda}{\varepsilon_{CONS}}\right)\right)^{(\dagger)}$

Table 1: Summary of algorithmic results, see Section 4 for details on notation. The table compares our new upper bounds to (known and new) lower bounds. Known lower bounds are graph-based and transfer to our setting, as we use weaker assumptions. To compare graph-based lower bounds with our upper bounds, one might choose $\Lambda = n$ to account for possible high degree and choose $\varepsilon_{SMB} = \varepsilon_{MMB} = n^{-c}$ to achieve w.h.p. correctness. (*) Lower bound proven in this paper using absMAC assumptions of [37]. (†) Lower bounds require runtimes of global broadcast to depend on n even though we perform a local analysis. (‡) Combinations of lower bounds of [2, 20, 42] for graph based models. (+) Trivial lower bound (Remark 5.3).

rather weak. We do neither require ability to measure signal strength, nor carrier sensing, nor synchronous wakeup nor knowledge of positions. We do assume, e.g., (arbitrary) bounds on the minimal physical distance between nodes and on the background noise (from which Λ can be derived), as well as conditional wakeup.

2.1 Comparison of Algorithmic Results with Previous Work

Global single-message broadcast. Table 2 compares the runtime of our algorithm for global SMB with previous work. Currently [14] and [32] provide the best implementations of global SMB in the SINR model (see the runtimes in Table 2). The result of [14] is as good or better than [32] in case $\log^{\alpha+1}(\Lambda) \le \log(n)$ and vice versa. To make it possible to compare our result to theirs, we need to choose $\varepsilon_{SMB} = 1/n^c$ such that global SMB is correct w.h.p.. Furthermore, we execute our algorithm with $\varepsilon' := \varepsilon/2$ instead of ε, while algorithms in previous work are executed without changing ε. This ensures that our bounds are stated in terms of the same parameter $D_{G_{1-\varepsilon}}$ rather than the possibly larger parameter $D_{G_{1-2\varepsilon}}$. At the same time the choice of ε' affects the runtime only by a constant factor. This results in a runtime of our algorithm of $\mathcal{O}\left((D_{G_{1-\varepsilon}} + \log(n))\log^{\alpha+1}(\Lambda)\right)$ in the strong connectivity graph $G_{1-\varepsilon}$. This improves over the algorithm presented in [14] in the full range of all parameters, and improves in case of $\log^{\alpha+1}(\Lambda) \le \min(D_{G_{1-\varepsilon}}\log(n), \log^2(n))$ over the algorithm of [32]. Note that compared to [32] we (and [14]) assume knowledge of a bound on Λ. The key-ingredient of this improvement is our localized analysis in combination with [37].

Article	Runtime bound for global SMB	We improve this runtime in case of
this	$\mathcal{O}\left((D_{G_{1-\varepsilon}} + \log(n)) \log^{\alpha+1}(\Lambda)\right)$	
[14]	$\mathcal{O}\left(D_{G_{1-\varepsilon}} \log^{\alpha+1}(\Lambda) \log(n)\right)$	all parameters and ranges
[32]	$\mathcal{O}\left(D_{G_{1-\varepsilon}} \log^2(n)\right)$	$\log^{\alpha+1}(\Lambda) \leq \min(D_{G_{1-\varepsilon}} \log(n)\ ,\ \log^2(n))$

Table 2: Comparison of the runtime of our global SMB protocol with previous results.

Global multi-message broadcast. The algorithm for global MMB derived from [29] runs in $\mathcal{O}((D_{G_{1-\varepsilon}} + k)(\Delta_{G_{1-\varepsilon}} \cdot \log n + \log^2 n))$ time. Roughly speaking, our algorithm replaces the dependency on the potentially large multiplicative term $D_{G_{1-\varepsilon}}\Delta_{G_{1-\varepsilon}}$ by $D_{G_{1-\varepsilon}}$ up to polylog factors. Section 3 summarizes global MMB in related models.

Global consensus. We are not aware of any previous work in the model we consider.

2.2 A Demonstration how Algorithms Benefit from Abstract MAC Layers

When abstract MAC layers were introduced to decompose global broadcast into local and global parts, the original goal was to understanding broadcast better and to achieve a general framework that can be used to state, implement and analyze new algorithms faster and simpler with respect to different models. A downside was that decomposing broadcast by adding a MAC layer might slow down performance. We demonstrate that the absMAC not only help to decompose the SINR-algorithm for global single-message broadcast of [14] into a local and global layer, but can be used to improve performance in an organized way when the algorithms of the two layers are modified and put back together. The key insight is, that the MAC layer provides the basic structure for a localized analysis by decomposing broadcast into a local and a global part. We show that a local analysis is indeed possible despite global interference and SINR constraints. To achieve best results, we make our analysis dependent on 1) local parameters such as the degree of a node, and 2) the desired probabilities of success of local broadcast. Combined with the algorithm [37] for global single-message and multi-message broadcast (that assumes an absMAC implementation such as ours), this immediately implies improved algorithms as highlighted in Section 2.1.

3 Related Work

Graph Based Wireless Networks. This model was introduced by Chlamtac and Kutten [7], who studied deterministic centralized broadcast. **Global SMB:** For the case where the topology is not known, Bar-Yehuda, Goldreich, and Itai (BGI) [4] provided a simple, efficient and fully distributed method called Decay for local broadcast. Using this method they perform global SMB in $\mathcal{O}(D \log n + \log^2 n)$ rounds w.h.p.. Later Czumaj and Rytter [12] and Kowalski and Pelc [40] simultaneously and independently presented an algorithm that performs global SMB in time $\mathcal{O}(D \log(n/D) + \log^2 n)$, w.h.p.. While this sequence of upper bounds was published, a lower bound of $\Omega(\log^2 n)$ was established for constant diameter networks by Alon et al. [2] and a lower bound of $\Omega(D \log(n/D))$ was established by Kushilevitz and Mansour [42]. Therefore the upper bounds are tight. In case the topology is unknown but collision detection is available, Ghaffari et al. [21], present show how to perform global SMB w.h.p. in time $\mathcal{O}(D + \log^6 n)$. In case the topology is known, a sequence of articles presented increasingly tighter upper bounds [8, 18, 16, 19, 39], where an algorithm for for global SMB in optimal time $\mathcal{O}(D + \log^2 n)$ was presented by Kowalski and Pelc [39]. **Global MMB:** When collision detection is available, the sequence of work [5, 21, 36] led to an $\mathcal{O}(D+k \log n+\log^2 n)$ round algorithm that performs global broadcast of k messages w.h.p. assuming knowledge of the topology, which is due to Ghaffari et al. [21]. When this assumption is removed, the runtime of [21] increases slightly to $\mathcal{O}(D+k \log n+\log^6 n)$. Earlier, Ghaffari et al. [20] showed a lower bound of $\Omega(k \log n)$ for global MMB. **Global Consensus:** Peleg [45] provided a good survey on consensus in wireless networks. Of particular interest is the work of Cholker et al. [9] and [1]. Many of these lower bounds can be transferred to the SINR-model using SINR-induced graphs. We can use upper bounds to benchmark our algorithms.

Abstract MAC layer. The abstract MAC layer model was recently proposed by Kuhn et al. [41]. This model provides an alternative approach to the various graph-based models mentioned above with the goal of abstracting away low level issues with model uncertainty. The probabilistic abstract MAC layer was defined by Khabbazian et al. [37]. **Implementations of absMACs:** Basic implementations of a probabilistic absMAC were provided by Khabbazian et. al [37] using Decay, and by [38] using Analog Network Coding. **Applications of absMACs:** The first to study an advanced problem using the absMAC of [41] were Cornejo et al. [10, 11], who investigated neighbor discovery in a mobile ad hoc network environment. Global SMB and MMB broadcast were studied by [37] in probabilistic environments and by Ghaffari et al. [23] in the presence of unreliable links. Newport [44] showed how to achieve fast consensus using absMAC implementations. Our paper makes applies the results of [37, 44].

SINR model. Moscibroda and Wattenhofer [43] were the first to study worst-case analysis in the SINR model. They pointed the algorithmic and distributed computing community to this model that was studied by engineers for decades. **Local broadcast:** Short time after this, Goussevskaia et al. [24] presented two randomized distributed protocols for local broadcast assuming uniform transmission power and asynchronous wakeup. This was improved simultaneously and independently by Yu et al. [48] and Halldorsson and Mitra [29] by obtaining similar bounds while using weaker model assumptions that are similar to those assumptions that we use. Both stated an algorithm for local broadcast in $\mathcal{O}(N_x \cdot \log(n) + \log^2(n))$, where N_x is the contention in the transmission range of node x. In this paper we transform the latter result to be part of an implementation of a probabilistic absMAC that yields fast acknowledgments. We modify the analysis of [29] to use purely local parameters. **Global MMB:** The above algorithms for local broadcast immediately imply algorithms with runtime $\mathcal{O}((D_{G_{1-\varepsilon}} + k)(\Delta_{G_{1-\varepsilon}} \cdot \log(n + k) + \log^2(n + k)))$ for global MMB of k messages w.h.p.. Scheideler et. al [46] consider a model with synchronous wakeup, uniform power and physical carrier sensing (that allows to differentiate signal strength corresponding to two thresholds). In this model they provide a randomized distributed algorithm that computes a constant density dominating set w.h.p. in $\mathcal{O}(\log n)$ rounds. Such a sparsified set can be used to speed up global MMB by replacing the dependency on $\Delta_{G_{1-\varepsilon}}$ in the formula above by $\log n$. Yu et al. [49, 50] obtain almost optimal bounds using arbitrary power control. For a large range

of the parameters their runtimes are better than the runtime of the algorithm that we provide. However, we point out that arbitrary power control is known to be almost arbitrarily more powerful for some problems than the uniform power restriction that we use [34, 43] such that we do not use this result as a benchmark. Power control was also used in [6] to achieve connectivity and aggregation, which in turn can be used for broadcast as well.
Global SMB: This problem recently caught increased attention and was studied in a sequence of papers [14, 31, 32, 33] using strong connectivity graphs $G_{1-\varepsilon}$. Jurdzinski, Kowalski et al. [31, 33] considered a setting where nodes know their own positions. In [31] they were able to present a distributed protocol that completes global broadcasts in the near-optimal time $\mathcal{O}(D + \log(1/\delta))$ with probability at least $1 - \delta$. In [33] they perform broadcasts within $\mathcal{O}(D \log^2 n)$ rounds. Daum et al. [14] propose a model that avoids the rather strong assumption that node's locations are known and does not use carrier sensing. However, they assume polynomial bounds on n and Λ. Thanks to a completely new approach they show how to still perform global broadcast in $G_{1-\varepsilon}$ within $\mathcal{O}(D \log^{\alpha+1}(\Lambda) \log(n))$ rounds w.h.p. using this weaker model. Their algorithm is based on a new definition of probabilistic SINR induced graphs combined with an iterative sparsification technique via MIS computation. We transfer and modify this algorithm to implement approximate progress in a probabilistic absMAC and provide a significantly extended analysis. Shortly after that, Jurdzinski et al. [32] came up with a $\mathcal{O}(D \log(n) + \log^2 n)$ algorithm that w.h.p. performs global broadcast independent of knowing Λ. However, to achieve this runtime they assume all nodes are awake and start the protocol at the same time. When assuming conditional wakeup, as [14] and we do, their algorithm still requires only $\mathcal{O}(D \log(n) + \log^2 n)$ rounds. Table 2 compares these results to ours.
Further work: During the last years significant progress was made on lower-level problems that might provide useful tools for absMAC design, such as connectivity [28], minimizing schedules [27], and capacity maximization [26, 34].

4 Model and Definitions

We begin by defining basic notation for graphs, which we use throughout the paper. Although the SINR model is not graph-based, we derive graphs from SINR models using reception zones. Abstract MAC layers are defined explicitly in terms of graphs. We continue by by describing the computational devices we use and recalling definitions of the SINR model, abstract MAC layers and global broadcast problems.

4.1 Graphs and their Properties

Let $G = (V, E)$ be a graph over n nodes V and edges E. We denote by $d_G(v, w)$ the hop-distance between w and v (the number of edges on a shortest (u, v)-path), and by $D_G := \max_{u,v \in V} d_G(u, v)$ the *diameter* of graph G. All neighbors of v in G are called *G-neighbors* of v. We denote the direct neighborhood of v in G by $N_G(v)$. This includes v itself. More formally we define $N_G(v) := \{u | (v, u) \in E\}$ and extend this to $N_{G,r}(v) := \{u | d_G(v, u) \leq r\}$ for the r-neighborhood, $r \in \mathbb{N}$. For any set $W \subseteq V$ we generalize this to $N_{G,r}(W) := \bigcup_{w \in W} N_{G,r}(w)$. The degree $\delta_G(v)$ of a node is the number of its (direct) neighbors in G, formally $\delta_G(v) := |N_G(v)| \setminus \{v\}$. We denote the maximum node degree of G by $\Delta_G := \max_{v \in V} \delta_G(v)$. Let $S \subseteq V$ be a subset of G's vertices, then $G|_S = (S, E|_S)$ denotes the subgraph of G induced by nodes S, where $E|_S := \{(u, v) \in E | u, v \in S\}$. A set $S \subseteq S' \subseteq V$ is called a *maximal independent set*

(MIS) of S' in G if 1) any two nodes $u, v \in S$ are independent, that is $(u, v) \notin E$, and 2) any node $v \in S'$ is covered by some neighbor in S, that is $N_G(v) \cap S \neq \emptyset$.

Definition 4.1 (Growth bounded graphs)**.** *A graph $G = (V, E)$ is* (polynomial) growth-bounded *if there is a polynomial bounding function $f(r)$ such that for each node $v \in V$, the number of nodes in the neighborhood $N_{G,r}(v)$ that are in any independent set of G is at most $f(r)$ for all $r \geq 0$.*

Lemma 4.2. *Let G be polynomially growth-bounded by function f, then it holds that $|N_{G,r}(v)| \leq \Delta f(r)$ for all $v \in V$ and $r \in \mathbb{N}$.*

Proof. The proof is deferred to Appendix A. □

4.2 The SINR Model

The following describes the foundations of the *physical model* (or *SINR model*) of interference. We start by introducing a second distance function. Nodes are located in a plane and we write $d(v, w)$ for the Euclidean distance between points v, w (often corresponding to node's positions). It is clear from the context when d refers to hop-distance or Euclidean distance.

When a node v (of a wireless network) sends a message, it transmits with (uniform) power $P > 0$. A transmission of v is received successfully at a node u, if and only if

$$SINR_u(v) := \frac{P/d(v,u)^\alpha}{\sum_{w \in S \setminus \{u,v\}} P/d(w,u)^\alpha + N} \geq \beta, \tag{1}$$

where N is a universal constant denoting the ambient noise. The parameter $\beta > 1$ denotes the minimum SINR (signal-to-interference-noise-ratio) required for a message to be successfully received, α is the so-called path-loss constant. Typically it is assumed that $\alpha \in (2, 6]$, see [24]. Here, S is the subset of nodes in V that are sending. (All other nodes send with power $P' = 0$). Independent of whether a distance function for the nodes is known, we assume in the analysis that the minimum distance between two nodes is 1[2] (a.k.a near-field effect). This assumption can be justified by scaling length when assuming that two nodes cannot be at the same position (e.g. as each antenna's size is strictly larger than 0). In this article we restrict attention to *uniform power* assignments. All nodes $v \in S$ send with the same power $P_v = P$ for some constant P. By $R := (P/\beta N)^{1/\alpha}$ we denote the transmission range, i.e. the maximum distance at which two nodes can communicate assuming no other nodes are sending at the same time. For $a \in \mathbb{R}^+$, we define $R_a := a \cdot R$. If $d(v, u) \leq R_a$ and $a < 1$, we say u and v are connected by a *a-strong* link. Like previous literature we consider a link to be strong if it is $(1 - \varepsilon)$-strong for constant $\varepsilon > 0$. Intuitively, this means that the link uses at least slightly more power than the absolute minimum needed to overcome the ambient noise caused, e.g., by a few nodes sending far away. If $R_a < d(u, v) \leq R_1$, we say u and v are connected by an *a-weak* link. A $(1 - \varepsilon)$-weak link is just called weak link. Strong connectivity is a reasonable and often used assumption [3, 14, 15, 24, 35].

[2]Otherwise the SINR-formula implies that the power $P/d(v, u)^\alpha$ received by a node u closer to a sender v is higher than the power that v uses to send.

4.3 SINR Induced Graphs

Like e.g. in [14], we consider *strong connectivity broadcast* in this article while using uniform power. Therefore we consider the strong connectivity graph $G_{1-\varepsilon} = (V, E_{1-\varepsilon})$, where $(u, v) \in E_{1-\varepsilon}$, if $u, v \in V$ are connected by a strong link. Given a graph G, we denote by Λ_G the ratio between the maximum and minimum Euclidean length of an edge in E. In case that G is $G_{1-\varepsilon}$, we simply write Λ instead of $\Lambda_{G_{1-\varepsilon}}$.

Remark 4.3. *Note that [14] uses a different but equivalent definition, while the above one is more common. In [14] two nodes are connected in a graph if they are at distance at most $R/(1+\rho)$, where ρ takes the place of our ε. When restating their lemmas, we simply use our notation without further comments as one can chose e.g. $\rho := \varepsilon/(1-\varepsilon)$ or $\varepsilon := \rho/(1+\rho)$.*

4.4 Abstract MAC Layers

While there are several abstract MAC layer models [23, 37, 41], the probabilistic version defined in [37] is most suitable for our purposes. Like any abstract MAC layer, the probabilistic MAC layer is defined for a graph $G = (V, E)$ and provides an acknowledged local broadcast primitive for communication in G. In our setting we are interested in strong connectivity broadcast with respect to the SINR formula, such that we use $G_{1-\varepsilon}$ as the communication graph (defined in Section 4.2).

We use the definitions of Ghaffari et al. [23] adapted to the probabilistic setting of [37]. To initiate such a broadcast, the MAC layer provides an interface to higher layers via input $bcast(m)_i$ for any node $i \in V$ and message $m \in M$. To simplify the definition of this primitive, assume w.l.o.g. that all local broadcast messages are unique. When a node $u \in V$ broadcasts a message m, the model delivers the message to all neighbors in E. It then returns an acknowledgment of m to u indicating the broadcast is complete, denoted by $ack(m)_u$. In between it returns a $rcv(m)_v$ event for each node v that received message m. This model provides two timing bounds , defined with respect to two positive functions, f_{ack} and f_{prog} which are fixed for each execution. . The first is the acknowledgment bound, which guarantees that each broadcast will complete and be acknowledged within f_{ack} time. The second is the progress bound, which guarantees the following slightly more complex condition : fix some $(u, v) \in E$ and interval of length f_{prog} throughout which u is broadcasting a message m; during this interval v must receive some message (though not necessarily m, but a message that some location is currently working on, not just some ancient message from the distant past). The progress bound, in other words, bounds the time for a node to receive some message when at least one of its neighbors is broadcasting. In both theory and practice f_{prog} is typically much smaller than f_{ack} [37]. Further motivation and power of these delay bounds is demonstrated e.g. in [23, 37, 41].

We emphasize that in abstract MAC layer models the order of receive events is determined non-deterministically by an arbitrary message scheduler. The timing of these events is also determined nondeterministically by the scheduler, constrained only by the above time bounds.

The Standard Abstract MAC Layer. Nodes are modeled as event-driven automata. While [23] assumes that an environment abstraction fires a wake-up event at each node at the beginning of each execution, we assume conditional wake-up to be consistent with the model of [14], see Definition 4.4. This is a weaker wake-up assumption with respect to upper bounds when compared to synchronous wake-up [23]. This strengthens our algorithmic results. In contrast to this our lower bounds assume synchronized wake-up, which is in turn the weaker assumption with respect to lower bounds. The environment is also responsible for any events specific to the problem being solved. In multi-message broadcast, for example, the environment provides the broadcast messages to nodes at the beginning.

Definition 4.4 (Conditional (a.k.a non-spontaneous) wake-up of [14] adapted to absMACs)**.** *Only after a node is woken up it can participate in computations below the MAC layer (i.e. in the network layer). Communication needs to be scheduled from scratch. This corresponds to the conditional (non-spontaneous) wake up model.*

The Enhanced Abstract MAC Layer. The enhanced abstract MAC layer model differs from the standard model in two ways. First, it allows nodes access to time (formally, they can set timers that trigger events when they expire), and assumes nodes know f_{ack} and f_{prog}. Second, the model also provides nodes an abort interface that allows them to abort a broadcast in progress.

The Probabilistic Abstract MAC Layer. We use parameters ε_{prog} and ε_{ack} to indicate the error probabilities for satisfying the delay bounds f_{prog} and f_{ack}. Roughly speaking this means that the MAC layer guarantees that progress is made with probability $1-\varepsilon_{prog}$ within f_{prog} time. With probability $1-\varepsilon_{ack}$ the MAC layer correctly outputs an acknowledgment within f_{ack} time steps. More details can be found in Section 4.2 of [37].

Reliable Communication. Note that like in [37] all our communication graphs $G := G_{1-\varepsilon}$ are static and undirected. In contrast to this, [23, 41] defines not only a graph G for guaranteed communication, but also a graph G' for possible (but unreliable) communication.

4.5 Problems

We derive algorithms in the SINR-model that perform the tasks listed below correctly with probability $1-\varepsilon_{task}$. When choosing $\varepsilon_{task} \leq n^{-c}$ we say that an algorithm performs a task with high probability (w.h.p.). Here, $c > 0$ is an arbitrary constant provided to the algorithm as an input-parameter. We use the notation w.h.p. only to compare our results with previous work.

The Multi-Message Broadcast Problem (MMB) [37]. This problem inputs $k \geq 1$ messages into the network at the beginning of an execution, perhaps providing multiple messages to the same node. We assume k is not known in advance. The problem is solved once every message m, starting at some node u, reaches every node in G. Note that we assume G is connected to be consistent with previous work in the SINR model, while in [23] this is not assumed. We treat messages as black boxes that cannot be combined.

The Single-Message Broadcast Problem (SMB) [37]. The SMB problem is the special case of MMB with $k = 1$. The single node at which the message is input is denoted by i_0.

The Consensus Problem (CONS), version considered in [44]. In this problem each node begins an execution with an initial value from $\{0, 1\}$. Every node has the ability to perform a single irrevocable *decide* action for a value in $\{0, 1\}$. To solve consensus, an algorithm must guarantee the following three properties: 1) *agreement*: no two nodes decide different values; 2) *validity*: if a node decides value v, then some node had v as its initial value; and 3) *termination*: every non-faulty process eventually decides.

4.6 General Model Assumptions

In principle each node has unlimited computational power. However, our algorithms perform only very simple and efficient computations. Finally, we assume that each node has private access to an unlimited perfect random source. This assumption can be weakened. As in [14] wake up of nodes is conditional, see Definition 4.4. From SINR-based work [14] that we use we take the following assumptions: Nodes are located in the Euclidean plane[3] and locations are unknown. Nodes send with uniform power, where the fixed power level P is not known to the nodes. We use the common assumption that $\alpha > 2$, see [24]. No collision detection mechanism is provided. Even when the same message is sent by (at least) two nodes and arrives with constructive interference we assume that the received signal cannot be distinguished from the case where no node is sending at all. As previous work we assume $G_{1-\varepsilon}$ is connected. MAC-layer based work [37] requires us to assume that nodes can detect if a received message originates from a neighbor in a graph G–in our setting this is $G_{1-\varepsilon}$–(only one graph G is used in [37], while messages from any sender in the network might arrive but do not cause rcv-events).

Remark 4.5 (Concerning SINR assumptions)**.** *Although not explicitly stated in [14], footnote 5 in the full version [13] of [14] indicates that they use the assumption $\alpha > 2$ as well. We also assume that α, β and N are known. Note that [14] allows α to be unknown, but fix in a known range $[\alpha_{\min}, \alpha_{\max}]$. Furthermore they assume upper and lower bounds for β and N given by $\beta_{\min}, N_{\min}$ and $\beta_{\max}, N_{\max}$. For simplicity we do not make these assumptions, but claim our results can be stated in terms of these bounds as well. Compared to [14] nodes that execute local broadcast do not need to know a polynomial bound on the network size n. In [14] this knowledge is only needed to achieve w.h.p. successful transmissions at each step. In our setting the desired probability of success is provided by the user of the absMAC.*

Remark 4.6 (Concerning absMAC assumptions)**.** *We want to remark that the assumption that nodes can detect if a received message originated in the $G_{1-\varepsilon}$-neighborhood is not used by any of the algorithms presented in this paper. In particular this assumption is not needed by previous algorithms on top of the MAC layer we use. This is due to the assumption that $G_{1-\varepsilon}$ is connected. However, being able to detect if a message was sent from a $G_{1-\varepsilon}$-neighbor might be required by future algorithms using our absMAC implementation that need broadcast to be implemented on exactly $G_{1-\varepsilon}$. Examples of this include $G_{1-\varepsilon}$-specific problems not studied in this article (such as e.g. computing shortest paths in $G_{1-\varepsilon}$). Note that a node x executing our absMAC implementation might also successfully transmit messages to nodes that are not in $N_{G_{1-\varepsilon}}(x)$ but still in transmission range. For the reasons explained above we state our algorithms without the assumption that nodes can detect in which range a received*

[3]Our results can be generalized to any growth-bounded metric space when revising the assumption on α.

message originated. If future algorithms using our absMAC implementation require exact broadcast, nodes executing our absMAC implementation could simply disregard messages they receive from nodes that are not their $G_{1-\varepsilon}$-neighbors (using the then necessary assumption that nodes can detect in which range a received message originated required to achieve exact local broadcast). If needed, there are several ways to implement this assumption. E.g. assuming that the SINR of the received message as well as the total received signal strength CCA can be measured. Using these assumptions there might also be faster SINR-implementations of an absMAC than provided in this article.

4.7 Overview of Frequently used Notation

For the convenience of the reader the following table summarizes notation used frequently (or globally) in this article. Definitions of notation not listed here are stated nearby where it is used.

Notation	Explanation/Reference
Problems:	
SMB	Global Single-Message Broadcast
MMB	Global Multi-Message Broadcast
CONS	Global Consensus
$\varepsilon_{SMB}, \varepsilon_{MMB}, \varepsilon_{CONS}$	Bounds on the probability that SMB,MMB and CONS are performed incorrectly
SINR related:	
α	$\alpha \in (2,6]$, path-loss exponent in the SINR model
P, P_v	Constant sending power of a node
$d(u,v)$	Euclidean distance between nodes u and v
$I_S(v)$	Interference caused at point v by nodes in set S sending with power P
ε	Parameter that helps to define strong reachability, see $R_{1-\varepsilon}$
R	$R := (P/\beta N)^{1/\alpha}$, transmission range if no other node sends (weak reachability)
$R_{1-\varepsilon}, R_{1-2\varepsilon}$	$R_a := a \cdot R$, transmission distance tolerating interference from a sparse set of nodes. Tolerated sparsity of the set depends on a (strong reachability)
Λ	Ratio of $R_{1-\varepsilon}$ to the shortest distance between any two nodes
Graph related:	
G_1	Nodes at distance at most R_1 are connected (weak connectivity graph). In our algorithms communication in this graph might be unreliable
$G_{1-\varepsilon}, G_{1-2\varepsilon}$	Nodes at distance at most $R_{1-\varepsilon}$/or $R_{1-2\varepsilon}$ are connected in $G_{1-\varepsilon}$/or $G_{1-2\varepsilon}$ (strong connectivity graphs). We implement reliable local broadcast in $G_{1-\varepsilon}$ and analyze fast approximate progress with respect to $G_{1-2\varepsilon}$
$G\|_S$	The subgraph $(S, E\|_S)$ of $G = (V,E)$ induced by nodes in S and $E\|_S := E \cap (S \times S)$
$N_G(v)$, $N_G(W)$	Neighborhoods of node v/set W in graph G
$N_{G,r}(v), N_{G,r}(W)$	r-neighborhood of node v/set W in graph G
Δ_G	Maximal degree of any node in graph G
D_G	Diameter of Graph G
f	Polynomial increasing function bounding the growth of graphs in this article
MAC layer related:	
$bcast, rcv, ack$	Broadcast/receive/acknowledgment events in the MAC layer
$f_{ack}, f_{prog}, f_{approg}$	Bounds on the time needed for acknowledgment/progress/approximate progress
$\varepsilon_{ack}, \varepsilon_{prog}, \varepsilon_{approg}$	Bounds on the probability that acknowledgment/progress/approximate progress are not performed in time $f_{ack}, f_{prog}, f_{approg}$
G'	Graph with unreliable communication [23]. Our setting considers $G' := G_1$

G	Graph with reliable communication [23]. Our setting considers $G := G_{1-\varepsilon}$
$\tilde{G}$	Notation we introduce to denote subgraphs/approximations of G in which we measure approximate progress. Our setting considers $\tilde{G} := G_{1-2\varepsilon}$
Algorithm 9.1 related:	
Phase ϕ	Phases $\phi = 1, \ldots, \Phi$ are executed within an epoch
Epoch	Each epochperforms approximate progress with respect to $G_{1-2\varepsilon}$
Φ	$\Phi = \Theta(\log \Lambda)$, number of phases ϕ executed in an epoch
Q	$Q = \log^{\alpha} \Lambda$, parameter used to adjust transmission probabilities in Line 11
c	$c \in \mathbb{N}$ s.t. $c \log^* N$ bounds the MIS algorithm [47]'s runtime on IDs $\in [1, N]$
p	$p \in (0, 1/2]$, transmission-probability of a node (whenever not specified otherwise)
μ	$\mu \in (0, p)$, reliability-probability of an edge (whenever not specified otherwise)
T	$T = \Theta\left(\log\left(f(h_1)/\varepsilon_{approg}\right)/(\gamma^2\mu)\right)$, transmission-repetitions (when not changed)
m, m'	m is the bcast-message to be broadcast, m' is a bcast-message that is received
γ	$\gamma \in (0, 1)$, parameter used to define the approximation $\tilde{H}_p^{\mu}[S]$ of $H_p^{\mu}[S]$
h_ϕ, h'_ϕ	$h_\phi := h'_\phi := 1$, $h'_\phi := 3h_{\phi+1}$ and $h_\phi := h'_\phi + c\log^*(\Lambda/\varepsilon_{approg}) + 1$ for $1 \leq \phi < \phi$
$H_p^{\mu}[S]$	Graph defined in [14]: μ-reliable edges, nodes $\in S$ send with prob. p, see Section 9.2
$\tilde{H}_p^{\mu}[S]$	Graph computed in [14] to $(1-\gamma)$-approximate $H_p^{\mu}[S]$ w.h.p., see Section 9.2
$\tilde{\tilde{H}}_p^{\mu}[S]$	Graph computed in Section 9.2. Likely to $(1-\gamma)$-approximate $H_p^{\mu}[S]$ locally
S_1	Set of nodes with an ongoing broadcast at a given time
S_ϕ	Independent Set in $\tilde{\tilde{H}}_p^{\mu}[S_\phi]$ that is (ϕ, i)-locally maximal with some probability
(ϕ, i)-locally	(ϕ, i)-locally maximal independent sets are defined in Definition 10.6
u_ϕ	Closest node $\in S_\phi$ to location i in the proofs, see Lemma 10.16
$U_{\phi,i}$	$U_{\phi,i} := N_{G_{1-\varepsilon}}(i) \cap S_\phi$, phase-$\phi$-senders at distance $\leq R_{1-\varepsilon}$, see Definition 10.5
$S_{\phi,i}$	$S_{\phi,i} := N_{\tilde{\tilde{H}}_p^{\mu}[S_\phi], h_\phi}(U_{\phi,i})$, phase-$\phi$-senders relevant to location i, see Definition 10.5
$S'_{\phi,i}$	$S'_{\phi,i} := N_{\tilde{\tilde{H}}_p^{\mu}[S_\phi], h'_\phi}(U_{\phi,i}) \subseteq S_{\phi,i}$, see Definition 10.5
W	Nodes that computed "wrong" neighbors in some $\tilde{\tilde{H}}_p^{\mu}[S_\phi]$, see Definition 10.2

5 Efficient Acknowledgments with an Application to Consensus

Theorem 5.1. *In the SINR model using the assumptions of Section 4.6, acknowledgments of an absMAC can be implemented w.r.t. graph $G_{1-\varepsilon}$ with probability guarantee $1 - \varepsilon_{ack}$ in time $f_{ack} = \mathcal{O}\left(\Delta_{G_{1-\varepsilon}} \log\left(\frac{\Lambda}{\varepsilon_{ack}}\right) + \log(\Lambda)\log\left(\frac{\Lambda}{\varepsilon_{ack}}\right)\right)$.*

Remark 5.2. *This section only focuses on implementation of broadcast. Reactions to inputs from the MAC layer, such as bcast$(m)_i$, are handled in Section 11. Section 11 that presents the final implementation of our absMAC.*

Proof. The bound on f_{ack} can be derived by modifying Theorem 3 in [29] to local parameters. We do this in Appendix B. The bound follows when Theorem B.3 is applied with parameter $\tilde{N}_x := 4\Lambda^2$, which upper bounds the number of nodes in transmission range R_1 and thus the local contention. We derive our claim as the actual contention N_x is upper bounded by $\Delta_{G_{1-\varepsilon}}$. Note that the network only knows a polynomial bound on Λ, not on N_x nor $\Delta_{G_{1-\varepsilon}}$, which in turn are estimated by Algorithm B.1. Furthermore one simply needs to modify Algorithm B.1 to stop after f_{ack} rounds, as then the probability-guarantee is reached. Note that this behavior does not guarantee that no messages from nodes that are not $G_{1-\varepsilon}$-neighbors are received. See Remark 4.6 how exact local broadcast can be implemented.

□

Remark 5.3. *As a node can only receive one message at a time, the degree $\Delta_{G_{1-\varepsilon}}$ of the network corresponds to the maximal contention at some node and is therefore a lower bound for f_{ack}. Therefore the result on f_{ack} in this section is close to optimal.*

5.1 Application to Network-Wide Consensus in the SINR Model

Before implementing the absMAC specification in a formal way in Section 11, we now derive an algorithm for global consensus based on the bound for acknowledgments, see Theorem 5.1. This serves as a first example to demonstrate the power of the absMAC theory when applied to the SINR world. This example is possible due to a result of [44] for achieving consensus using an absMAC using the fact that they analyze this problem in terms of f_{ack}, while f_{prog} does not appear in their runtime. Although the MAC layer used in [44] is deterministic, we can obtain a randomized algorithm that works correct with probability $1-\varepsilon_{CONS}$ by choosing ε_{prog} and ε_{ack} to be at most $\frac{\varepsilon_{CONS}}{t(n)n^2}$, where $t(n)$ is the runtime of the algorithm using the MAC layer. We obtain the following Theorem.

Theorem 5.4 (Theorem 4.2 of [44] transferred to our setting)**.** *The wPAXOS algorithm [of [44]] solves network-wide consensus in $\mathcal{O}(D_{G_{1-\varepsilon}} \cdot f_{ack})$ time in the (probabilistic) absMAC model (with $\varepsilon_{ack} = \varepsilon_{prog} = \frac{1}{n^4 \varepsilon_{CONS}}$) in any connected network topology $G_{1-\varepsilon}$ w.h.p., where nodes have unique ids and knowledge of network size.*

Plugging in the bounds on f_{ack} of Theorem 5.1 we obtain:

Corollary 5.5 (Theorem 4.2. of [44] transferred to our setting)**.** *Network-wide consensus can be solved with probability $1-\varepsilon_{CONS}$ in time*

$$f_{CONS} = \mathcal{O}\left(D_{G_{1-\varepsilon}}(\Delta_{G_{1-\varepsilon}} + \log(\Lambda))\log\left(\frac{n\Lambda}{\varepsilon_{CONS}}\right)\right).$$

6 Impossibility of Fast Progress using the SINR-Model

Many algorithms that are implemented in an absMAC benefit from the fact that typically f_{prog} is much smaller than f_{ack}. Often it is the case that $f_{prog} = \mathcal{O}(\text{polylog}(f_{ack}))$. We show that for any implementation of the absMAC [37] for $G_{1-\varepsilon}$ in the SINR model such a difference of the runtime is impossible. One can even not expect a bound on f_{prog} that is much better than f_{ack}. As the bound on f_{ack} in Theorem 5.1 is close to our lower bound on f_{prog}, we conclude that this algorithm is an almost optimal implementation of absMAC in the SINR-model with respect to both f_{ack} and f_{prog}.

Theorem 6.1. *For worst-case locations of points there is no implementation of the absMAC in the SINR model that provides local broadcast in $G_{1-\varepsilon}$ and achieves fast progress. In particular it holds that $f_{prog} \geq \Delta_{G_{1-\varepsilon}}$. This is true even for an optimal schedule computed by an (even central) entity that has unbounded computational power, has full knowledge as well as control of the network and can choose an arbitrary power assignment.*

Proof. We first recall a slightly more formal definition of f_{prog} given in [37], Section 4.1. (we can choose $G = G'$ in their notation, as we do not consider unreliable links): *a $rcv(m)_j$ event can only be caused by a $bcast(m)_i$ event when the proximity condition $(i,j) \in E$ is satisfied. The progress bound guarantees, that a rcv event occurs at j within time f_{prog} when some neighbor of j is broadcasting some message.* However, we make use of the assumption made in [37] that nodes perform only *rcv* events for messages they received from $G_{1-\varepsilon}$-neighbors and discard messages from other nodes in transmission range. (See Remark 4.6 for a discussion on this and why this assumption is later not needed for our upper bounds with respect to global broadcast implementations.)

The reader might like to consult Figure 1 while following our construction. For simplicity we consider the Euclidean setting and the implied distances. Consider Δ nodes $V := \{v_1, \ldots, v_\Delta\}$ placed equidistant on a line with distance 1 between neighboring nodes. Let $R_{1-\varepsilon} := 10\Delta$, i.e. parameters N, P and β are chosen such that the transmission range is 10Δ, and consider a second line parallel to the first line at distance $R_{1-\varepsilon}$ to the first line. Now assume Δ nodes $U := \{u_1, \ldots, u_\Delta\}$ are placed equidistant on this second line with distance 1 between neighboring nodes. Each node in V and U has degree $\Delta_{G_{1-\varepsilon}} = \Delta$ in $G_{1-\varepsilon}$. Each node in V has exactly one edge in $G_{1-\varepsilon}$ to a node in U and vice versa. W.l.o.g. we assume v_i is connected to u_i. Now assume that $bcast(m_v)_v$ events occurred for each $v \in V$ and a message $m_v \in M$. Due to SINR constraints, a transmission through edge (v_i, u_i) is only successful when no other node in $V \cup U \setminus \{v_i\}$ is sending at the same time. Although u_i receives a message from u_i in case of a successful transmission, no node in $U \setminus \{u_i\}$ receives a message. As there are $\Delta_{G_{1-\varepsilon}}$ pairs (v_i, u_i), there is a node u_j that does not receive a message from a $G_{1-\varepsilon}$-neighbor during the first $\Delta_{G_{1-\varepsilon}} - 1$ time slots. Because u_j has a neighbor that is broadcasting, we conclude that $f_{prog} \geq \Delta_{G_{1-\varepsilon}}$ for any algorithm. □

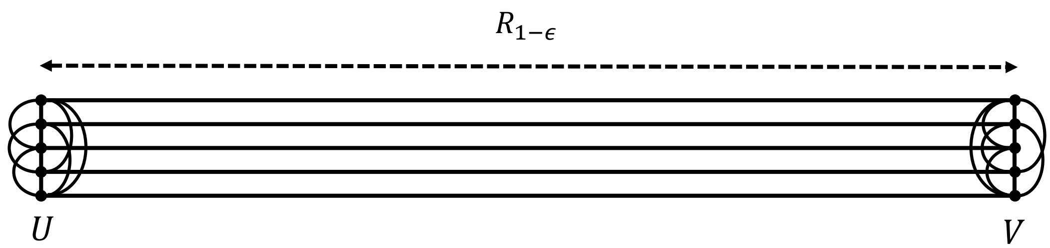

Figure 1: Graph $G_{1-\varepsilon}$ based on the construction used in the proof of Theorem 6.1. Here we choose $\Delta = 5$.

Remark 6.2 (Comparison with previous lower bounds in the SINR model)**.** *Somewhat similar arguments were made earlier in [14]. The same lower bound on f_{prog} might also be derived by modifying the construction in the proof of Theorem 9 in [14] to our setting when transferring it to the notion of f_{prog}. However, their lower bound is on the runtime for global single-message broadcast in (the weak connectivity) graph G_1 and therefore would need to be adapted.*

Remark 6.3 (Comparison with previous lower bounds in absMACs)**.** *Note that in [22], Theorems 7.1 and 7.2 also provide a lower bound of $\Omega(\Delta \log n)$ for f_{prog} in the dual-graph model with unreliable links. The construction of their lower bound has a similar flavor to* *ours: Graph G consists of Δ edges between nodes in U and V like in our example. Graph G' is the complete bipartite graph over U and V. Whenever one single node in U is sending, an adversary prevents communication through unreliable edges (such that progress is only made at one node). When more than one node in U is sending, the adversary allows communication through unreliable edges in a way that causes the lower bound. The difference to our model is, that in our setting the edges in G' correspond to interference when a node incident to an edge is sending. This interference is fixed whenever a node is sending and cannot be switched on/off by an adversary. At the same time messages received via edges in G' that are not in G are discarded by the probabilistic MAC layer described in [37] such that no progress can be made using these edges.*

7 Approximate Progress

Due to the lower bound on progress of Section 6 we cannot expect f_{prog} to be much better than f_{ack} in any SINR implementation. However, like in other wireless models it should take much less time until some message is received by a node v (when several neighbors of v are sending) compared to the time it takes until all neighbors of a sending node u receive u's message. The problem might be that the absMAC specification tries to measure progress in this physical model using a definition of progress that tried to capture the whole complication of the SINR model by a single graph. Motivated by this we try to capture a sense of progress by using two graphs and modify the absMAC specification. An easy way would be to relax the progress bound and output a rcv-event not only for messages sent by $G_{1-\varepsilon}$-neighbors, but for all message received (i.e. sent by any G_1 neighbor). This is problematic when considering randomized algorithms. In particular when computing e.g. overlay networks. It might happen that only $G_1 \setminus G_{1-\varepsilon}$-neighbors of a node v are chosen for the overlay due to the random event of low interference. This could of course be avoided by directly implementing the absMAC with respect to G_1 rather than $G_{1-\varepsilon}$, which in turn results in a $\Omega(n)$ lower bound for f_{prog} and f_{ack} (e.g. when all nodes are located at distance at least R_1 such that messages can only be received when exactly one node is sending). Later these overlay nodes might not be able to serve v. To avoid such a setting, we introduce an *approximate* progress bound into the absMAC specification, where we use a graph G and an *approximation* (or any subgraph) $\tilde{G}$ of G in which progress is measured.

In the next sections we show that this generalization of progress has three desirable properties, it

1. captures SINR behavior in the sense that we present an absMAC implementation in the SINR model that provides fast (approximate) progress, and
2. replaces (with minor assumptions and effects) the progress bound in the runtime-analysis of e.g. global single-message and multi-message broadcast in the MAC layer [37], and
3. does not affect the correctness of these algorithms.

Therefore we consider this notion of approximate progress to be a good modification of the specification of abstract MAC layers with respect to the SINR model.

Definition 7.1 (Approximate progress)**.** *Let there be (reliable[4]) broadcast implemented with respect to a graph G and let $\tilde{G} := (V, \tilde{E})$ be a subgraph[5] of G. Consider a node i and assume that a $\tilde{G}$-neighbor of i is broadcasting a message. The approximate progress bound guarantees that a rcv event with a message originating in a G-neighbor occurs at node i within time f_{approg} with probability $1 - \varepsilon_{approg}$. We say that approximate progress is implemented with respect to graphs G and (its approximation) $\tilde{G}$.*

We formalize this using the notation of [37]: Let β be a closed execution that ends at time t. Let I be the set of $\tilde{G}$-neighbors of j that have active bcasts at the end of β, where $bcast(m_i)_i$ is the bcast at i. Suppose that I is nonempty. Let I' be the set of G-neighbors of j that have active bcasts at the end of β. Suppose that no $rcv(m_i)_j$ event occurs in β, for any $i \in I'$. Define the following sets A and B of time-unbounded executions that extend β.

- *A, the executions in which no $abort(m_i)_i$ occurs for any $i \in I$.*
- *B, the executions in which, by time $t + f_{approg}$, at least one of the following occurs:*
  1. *An $ack(m_i)_i$ for every $i \in I$,*
  2. *A $rcv(m_i)_j$ for some $i \in I'$, or*
  3. *A rcv_j for some message whose bcast occurs after β.*

If $\mathbb{P}_\beta[A] > 0$, then $\mathbb{P}_\beta[B|A] \geq 1 - \varepsilon_{approg}$.

This notation is useful, as there are settings where it is not crucial that progress is made with respect to exactly G. Already progress in subgraph $\tilde{G}$ might yield good overall bounds for solving a problem on G especially when e.g. (depending on the problem at hand) $D_{\tilde{G}} \approx D_G$ or $\tilde{G}^2$. As we show in Theorem 12.6, in the global SMB and MMB algorithms of [37] local broadcast does not need to be precise such that under some conditions progress can be replaced by approximate progress. In the global broadcast algorithms of [37], once a message is received by a node i, node i broadcasts the message if it did not broadcast it before. The result of global broadcast is independent of whether a message was received due to transmission from a $\tilde{G}$-neighbor or a G-neighbor. However, one still needs to consider f_{ack} with respect to G. At the same time this obervation allows us to express large parts of runtimes of global broadcast algorithms [37] in terms of $D_{\tilde{G}}$ and f_{approg} instead of D_G and f_{prog}. In graph based models, one could choose e.g. $G := \tilde{G}^2$, the graph that is derived from $\tilde{G}$ when all paths of length at most 2 in $\tilde{G}$ are replaced by edges. In the SINR model one might choose, e.g., $G := G_{1-\varepsilon} \supseteq G_{1-2\varepsilon} =: \tilde{G}$, as we do. This choice captures that any $G_{1-\varepsilon}$-neighbor is almost a $G_{1-2\varepsilon}$-neighbor. In addition its signal has a similar strength when it arrives at the receiver and in reality might even be the same, as signal strengths can vary slightly. The factor 2 in $1 - 2\varepsilon$ can be chosen arbitrarily small, but must be greater than 1.

Remark 7.2. *Like [23] we study a dual-graph model. However, in our setting all communication is reliable. They consider a setting where reliable communication is provided in G and communication which is unreliable in a non-deterministic way in $G' \supset G$. We extend this by a third graph $\tilde{G} \subset G$ such that unreliability can be studied in addition to approximate progress*

[4]The notation of approximate progress might later be extended to unreliable broadcast [23].

[5]Graph $\tilde{G}$ can be any subgraph of G but will typically be an approximation of G, which results in the name approximate progress. Later we consider graph $\tilde{G} := G_{1-2\varepsilon}$, which approximates $G := G_{1-\varepsilon}$ with respect to the SINR formula and Euclidean distances in the sense that it contains all, but the longest edges of G.

in future work. Note that approximate progress in $\tilde{G}$ inherits reliable communication from G. It is important to note that lower bounds on MMB of [23] in their gray-zone model *with unreliable graphs do not apply in our setting. Their lower bounds are invalid, as we consider reliable broadcast in $G \supset \tilde{G}$ and assume that G is connected.*

8 Decay Fails to Yield Fast Approximate Progress

Inspired by a proof of [14], we transfer this result into our setting. We show that using a (standard) Decay method, one cannot achieve fast approximate progress in the SINR model. In Section 9 we present an implementation based on an algorithm of [14] that uses a different strategy than Decay achieves fast approximate progress and is analyzed in a more precise way.

Theorem 8.1. *When using the* Decay *method of [4] to implement local broadcast of a MAC layer in the SINR model, it holds that $f_{approg} = \Omega(\Delta_{G_{1-\varepsilon}} \log(1/\varepsilon_{approg}))$.*

Proof. In the (standard) Decay method of [4] for graph-based models with collision detection, each node starts with sending probability 1 and halves its transmission probability in each time slot until a sending probability is reached where no collision occurred for the first time. Then it keeps transmitting with this probability. This method can be applied in the SINR model as well (note that adding the assumption of collision detection yields a stronger lower bound than using our model assumptions).

Consider two $R_{1/4}$-balls whose centers are located at distance R_2. Let ball B_1 contain 2 nodes and let ball B_2 contain $\Delta_{G_{1-\varepsilon}} \leq n/2$ nodes. In the corresponding graph $G_{1-\varepsilon}$ the nodes located in different balls are not directly connected. We assume that the remaining $n/2-2$ nodes are arranged such that the nodes in the two balls are connected by a path of length $n/2-1$ in $G_{1-\varepsilon}$. Let's assume each of the nodes in B_1 and B_2 wants to broadcast a message and we perform a (standard) Decay mechanism. Once the probabilities reach a level where the nodes in B_1 are likely to transmit, the interference from nodes in B_2 is very strong. To be more formal, in round i, the probability that exactly one node in B_1 is sending is less than $2^{-(\log \Delta_{G_{1-\varepsilon}}-i)}$. The probability that no node in B_2 is sending is $(1-1/2^{\log(\Delta_{G_{1-\varepsilon}})-i})^{\Delta_{G_{1-\varepsilon}}} \leq e^{-(\Delta_{G_{1-\varepsilon}})/2^{\log(\Delta_{G_{1-\varepsilon}})-i}}$. Thus the success probability of a node in B_1 is at most $e^{-(\Delta_{G_{1-\varepsilon}})/2^{\log(\Delta_{G_{1-\varepsilon}})-i}}/2^{-\log(\Delta_{G_{1-\varepsilon}})+i}$. From this we conclude that the probability that a successful transmission takes place in B_1 within $\log(\Delta_{G_{1-\varepsilon}})$ rounds with $i=1,\ldots,\log(\Delta_{G_{1-\varepsilon}})$ is less than

$$\begin{aligned}
&\sum_{i=1}^{\log(\Delta_{G_{1-\varepsilon}})} e^{-(\log(\Delta_{G_{1-\varepsilon}}))/2^{\log(\Delta_{G_{1-\varepsilon}})-i}}/2^{-\log(\Delta_{G_{1-\varepsilon}})+i} \\
= &\sum_{i=\log(\Delta_{G_{1-\varepsilon}})-\log\log(\Delta_{G_{1-\varepsilon}})}^{\log(\Delta_{G_{1-\varepsilon}})} e^{-\log(\Delta_{G_{1-\varepsilon}})/2^i}/2^{-\log(\Delta_{G_{1-\varepsilon}})+i} \\
&+ \sum_{i=1}^{\log(\Delta_{G_{1-\varepsilon}})-\log\log(\Delta_{G_{1-\varepsilon}})} e^{-\log(\Delta_{G_{1-\varepsilon}})/2^i}/2^{-\log(\Delta_{G_{1-\varepsilon}})+i}
\end{aligned}$$

We bound this by

$$
\begin{aligned}
&\leq \quad e^{-\log(\Delta_{G_{1-\varepsilon}})/2^{\log\log(\Delta_{G_{1-\varepsilon}})}} \sum_{i=\log(\Delta_{G_{1-\varepsilon}})-\log\log(\Delta_{G_{1-\varepsilon}})}^{\log(\Delta_{G_{1-\varepsilon}})} 1/2^{-\log(\Delta_{G_{1-\varepsilon}})+i} \\
&\quad + \ e^{-\log(\Delta_{G_{1-\varepsilon}})/2^{\log(\Delta_{G_{1-\varepsilon}})-\log\log(\Delta_{G_{1-\varepsilon}})}} \sum_{i=1}^{\log(\Delta_{G_{1-\varepsilon}})-\log\log(\Delta_{G_{1-\varepsilon}})} 1/2^{\log(\Delta_{G_{1-\varepsilon}})-i} \\
&\leq \quad e^{-1/2}/2^{-\log(\Delta_{G_{1-\varepsilon}})-\log\log(\Delta_{G_{1-\varepsilon}})+1} + 2e^{-\log(\Delta_{G_{1-\varepsilon}})+1} \\
&\leq \quad c\frac{\log(\Delta_{G_{1-\varepsilon}})}{\Delta_{G_{1-\varepsilon}}} \text{ for some constant } c
\end{aligned}
$$

Therefore, $\left(c\frac{\log(\Delta_{G_{1-\varepsilon}})}{\Delta_{G_{1-\varepsilon}}}\right)^{-1}\ln(1/\varepsilon_{prog})$ repetitions of $\log(\Delta_{G_{1-\varepsilon}})$ rounds $i = 1, \ldots, \log(\Delta_{G_{1-\varepsilon}})$ are necessary such that the nodes in B_1 make progress with probability ε_{approg}. We conclude that $f_{approg} = \Omega(\Delta_{G_{1-\varepsilon}} \log(1/\varepsilon_{prog})$. □

Note that the authors of [14] presented a lower bound of $\Omega(n)$ for SMB in $G_{1-\varepsilon}$ (Theorem 8 of [14]), when using the Decay method of [4]. This $\Omega(n)$ lower bound is of interest, as the construction of [14] allows for an algorithm that needs only $\mathcal{O}(1)$ rounds for SMB. Looking more closely at their lower bound, this can be interpreted as $f_{approg} = \Omega(n)$ when $\varepsilon_{approg} = n^{-c}$. We strengthen this lower bound for f_{approg} to $f_{approg} = \Omega(n\log(1/\varepsilon_{approg}))$. In the proof of this lower bound we use that the SINR model takes global interference into account (in contrast to graph based models). Also note that the proof of Theorem 8 of [14] uses a network with maximal degree $\mathcal{O}(n)$, and it can be easily generalized to yield $f_{prog} = \Omega(\Delta_{G_{1-\varepsilon}})$ for arbitrary maximal degrees $\Delta_{G_{1-\varepsilon}}$.

9 Implementation of Fast Approximate Progress

Now we describe a method different from Decay . Note that during an execution of the implementation additional messages from nodes that are G_1-neighbors but not $G_{1-\varepsilon}$-neighbors might occur in a probabilistic way. These do not affect our delay bound for approximate progress with respect to $G_{1-2\varepsilon}$, as the analysis guarantees that messages from $G_{1-\varepsilon}$-neighbors arrive within time f_{approg}. Note that Remark 5.2 applies to this Algorithm as well. This Algorithm 9.1 is described in this section and analyzed in Section 10.

Theorem 9.1. *In the SINR model using the assumptions of Section 4.6, Algorithm 9.1 implements approximate progress of an absMAC with respect to graphs $G_{1-\varepsilon}$ and its approximation $G_{1-2\varepsilon}$ with probability at least $1-\varepsilon_{approg}$ in time approximate progress of an absMAC with respect to graphs $G_{1-\varepsilon}$ and its approximation $G_{1-2\varepsilon}$ with probability at least $1-\varepsilon_{approg}$ in time*

$$f_{approg} = \mathcal{O}\left(\left(\log^{\alpha}(\Lambda) + \log^{*}\left(\frac{1}{\varepsilon_{approg}}\right)\right)\log(\Lambda)\log\left(\frac{1}{\varepsilon_{approg}}\right)\right).$$

The algorithm presented by [14] achieves global SMB in the strong connectivity graph $G_{1-\varepsilon}$. We modify this algorithm to fast (probabilistic) approximate progress with respect to $G_{1-2\varepsilon}$. In the algorithm of [14], after a node receives a bcast-message, it immediately forwards this (uniform) bcast-message. Inspired by this we implement this part in a similar style. However, we handle the possibility of multiple bcast-messages and need to guarantee that fast approximate progress can be proven. However, the modifications of their algorithm are substantial as described in this section to make it suitable for a localized analysis. In particular, in order to get an improved time-bound, we need to 1) introduce non-unique temporary labels instead of using unique IDs and handle this non-uniqueness, 2) acknowledge certain messages involved in coordination below the MAC layer, and 3) reduce the number T of repeated transmissions such that T is just large enough to guarantee low expected global interference from parts of the plane where computations went into a wrong direction based on communication-mistakes due to the reduced number T of repetitions. The analysis in Section 10 uses several Lemmas from [14]. Whenever proofs of [14] do not need to be changed significantly, we state versions of them adapted to our setting in Appendix C.

9.1 High-Level Description

We start by presenting a high-level outline of the algorithm. We follow the approach of [14] and perform epochs, each consisting of f_{prog} time steps. Each epoch corresponds to Lines 6–15 of Algorithm 9.1. During each epoch we compute approximations of a sequence of constant degree graphs $H_1, H_2, \ldots, H_\Phi$, $\Phi = \Theta(\log \Lambda)$, used for communication. Graph H_1 is defined based on vertex set S_1, which is the set of nodes that have an ongoing broadcast at this time. As this set S_1 might change over time (depending on the algorithm using the absMAC and conditional wale-up, see Definition 4.4), graphs $H_1, H_2, \ldots, H_\Phi$ might be different in different epochs. Each H_ϕ, $\phi > 1$, is defined based on the nodes of a maximal independent set in $H_{\phi-1}$. For each H_ϕ it is guaranteed that, when each node in H_ϕ transmits with a certain (constant) probability $p \in (0, 1/2]$, then for each edge e of H_ϕ the transmission through e is successful with a (constant) probability $\mu \in (0, p)$. Using geometric arguments we show in Lemma 10.18 that when for Φ phases $\phi = 1, \ldots, \Phi$ during phase ϕ all nodes of graph H_ϕ transmit their message a certain amount of times, then approximate progress takes place in $G_{1-2\varepsilon}$ within time $f_{approg} = \mathcal{O}\left(\left(\log^{\alpha}(\Lambda) + \log^{*}\left(\frac{1}{\varepsilon_{approg}}\right)\right)\log(\Lambda)\log\left(\frac{1}{\varepsilon_{approg}}\right)\right)$. This happens with probability $1 - \varepsilon_{approg}$.

Intuition behind this algorithm: Intuitively, this algorithm automatically adapts to regions of varying density. As the vertex set of graph H_ϕ is an MIS of $H_{\phi-1}$, it is typically a sparser version (with respect to density of nodes in the plane) of $H_{\phi-1}$. Finally we show that H_Φ is so sparse that nodes are too far away to communicate due to SINR constraints. Due to this sparsity, each node at this level is able to broadcast a message to its $G_{1-\varepsilon}$-neighbor with some probability. During this algorithm it will turn out that for each node $u \in N_{G_{1-2\varepsilon}}(S_1)$, that has $G_{1-2\varepsilon}$-neighbor with an ongoing broadcast, there is a $G_{1-\varepsilon}$-neighbor of u in some S_ϕ from which u receives a message in phase ϕ. In particular, in phase ϕ the local density of nodes is reduced in a way that 1) there is still a node u_ϕ at distance at most $R_{1-\varepsilon}$, and 2) the density of nodes is so low that interference from these other nodes is low enough that u_ϕ's message reaches u with some probability (due to random transmissions which further sparsify the set of transmitting nodes). We modify and extend the algorithm and analysis of [14] and choose parameters of the algorithm to our benefit.

Suitability for localized analysis: Thanks to the MAC layer that helps us to treat global and local parts of an algorithm separately in a structured way, we only need to provide an algorithm that ensures local approximate progress in order to implement this part of the MAC layer, while the algorithm of [14] has to ensure global broadcast (and focuses on single-message broadcast, while we study multi-message broadcast). Note that therefore the authors of [14] need to ensure that all their iterative computations of global approximations of communication graphs H_ϕ have the desired approximation-quality with high probability in n. Compared to this, we only need to make sure that for any point i in space these graphs are local approximations with a certain probability. Therefore we require only a much lower probability and gain a speedup from this. In particular this probability only depends on the number of coordination-messages exchanged by those nodes that are locally involved in ensuring approximate progress of bcast-messages[6] that might reach i, as the sender is at distance at most $1-\varepsilon$ to i.

We make use of this locality aspect (in combination with the carefully chosen parameters) and perform a more careful and localized analysis extending the one of [14].

Naturally, some parts of our proof follow along the lines of the proof in [14] or argue how their proofs can be adapted. However, they are significantly extended to derive the speedup from our modifications of their algorithm. In the end our detailed analysis of approximate progress yields faster global SMB than [14], see Section 12.

9.2 Graphs

During the algorithm we consider a graph $H_p^\mu[S]$ that was defined in [14]. This graph depends on a set of nodes S, a constant transmission probability $p \in (0, 1/2]$ and a constant reliability parameter $\mu \in (0, p)$. The vertices of $H_p^\mu[S]$ are just the nodes in S. To define the edge set of $H_p^\mu[S]$, assume that each node in S sends with probability p and no node outside of S (i.e. in $V \setminus S$) is sending at the same time. Based on this assumption/experiment we define the edge set $E_p^\mu[S]$ to contain edge $(u, v) \in S \times S$ iff (i) u receives a message from v with probability at least μ, and (ii) v receives a message from u with probability at least μ.

As it is difficult to compute $H_p^\mu[S]$ in a distributed way (as pointed out in [14]), the authors of [14] compute a $(1-\gamma)$-approximation $\tilde{H}_p^\mu[S] = (S, \tilde{E}_p^\mu[S])$, where w.h.p. the following is true:

$$E_p^\mu[S] \subseteq \tilde{E}_p^\mu[S] \subseteq E_p^{(1-\gamma)\mu}[S].$$

To obtain a speedup, we do not compute graph $\tilde{H}_p^\mu[S]$, but define and compute a graph $\tilde{\tilde{H}}_p^\mu[S] = (S, \tilde{\tilde{E}}_p^\mu[S])$ that locally corresponds to $\tilde{H}_p^\mu[S] = (S, \tilde{E}_p^\mu[S])$ at each point i with some probability much smaller than w.h.p (and demonstrate later that this is enough for our purposes). We postpone the precise formal definition of this locality to Definitions 10.5 and 10.8. There we define local correctness with respect to different sets S_ϕ together with other requirements for correct local computation during the algorithm. By postponing the definition, we avoid unnecessary general and therefore complicated notation. For now we only need to know that it should always be the case that for any node v we desire that $N_{\tilde{\tilde{H}}_p^\mu[S]}(v)$ corresponds to neighbors of v that would be present in a $(1-\gamma)$-approximation $\tilde{H}_p^\mu[S]$ of

[6]We denote by *bcast-message* any messages that contains information to be broadcast due to an bcast-event. By *messages*, we refer to messages sent for coordination among the nodes.

$H_p^\mu[S]$ as well. However, we typically consider a much larger neighborhood of v and desire that the subgraph of $\tilde{\tilde{H}}_p^\mu[S]$ corresponding to this neighborhood matches the corresponding subgraph of a $(1-\gamma)$-approximation $\tilde{H}_p^\mu[S]$ of $H_p^\mu[S]$.

9.3 Details of the Algorithm

We propose the following algorithm that is executed by all nodes in S_1 and is inspired by [14], but has small modifications that yield substantial improvements when analyzed in detail. The algorithm consists of epochs that are continuously repeated and ensure approximate progress within each execution of a epoch. Like in [14] we assume that all nodes get synchronized by other nodes when they wake up and join the algorithm at the beginning of the next epoch. A node i wakes up either due to receiving a bcast-message from another node or due to the first *bcast* event that occurred at node i. Whenever a $bcast(msg)_i$ event occurs, a variable m stored in node i is set to msg.

Algorithm 9.1 Implementation of the part of absMAC that achieves fast approximate progress. As executed by a node i.

CONTINUOUS EXECUTION OF EPOCHS:
1: $\Phi := \Theta(\log \Lambda)$; $Q := \Theta(\log^\alpha \Re)$;
2: **while** awake **do**
3:     $S_1 := S$;
4:     **if** $m \neq 0$ **then** // ongoing broadcast of m
5:         node i marks itself as contained in set S_1;
6:         **for** $\phi = 1, \ldots, \Phi$ **do**
7:             **if** $i \in S_\phi$ **then**
8:                 Compute graph $\tilde{\tilde{H}}_p^\mu[S_\phi]$ and schedule τ_ϕ as described in Section 9.3.1;
9:                 Compute $S_{\phi+1}$ as described in Section 9.3.2;
10:                 **for** $\mathcal{O}(Q \cdot \log(1/\varepsilon_{approg}))$ rounds **do**
11:                     **transmit** bcast-message m with probability p/Q;
12:                     // If not transmitting, listen for a bcast-message
13:                 **end for**
14:             **end if**
15:         **end for**
16:     **end if**
17:     m':= (first) bcast-message received due to a transmission from another node in Line 11;
18:     **output** $rcv(m')_i$;
19: **end while**

Once a $bcast(msg)_i$ event occurs at node i at time t, we say that node i has an ongoing broadcast for $f_{ack}/2$ time steps starting at time $t+1$. At the beginning of each epoch, each node i marks itself as belonging to set S_1 if it has an ongoing broadcast (Line 5 of Algorithm 9.1). Whenever a node i receives a bcast-message m' for the first time in an epoch, it delivers that bcast-message to its environment with a $rcv(m')_i$ output event. This behavior does not guarantee that no messages from nodes that are not $G_{1-\varepsilon}$-neighbors are received. See Remark 4.6 how exact local broadcast can be implemented.

Next in the epoch, a sequence of sets $S_1 \supseteq S_2 \supseteq \cdots \supseteq S_\Phi$ and corresponding graphs $\tilde{\tilde{H}}_p^\mu[S_1] \supseteq \cdots \supseteq \tilde{\tilde{H}}_p^\mu[S_\Phi]$ are computed[7]. In this sequence $\tilde{\tilde{H}}_p^\mu[S_\phi]$ is a graph, that given any node $i \in S_1$, is likely to $(1-\gamma)$-approximate $H_p^\mu[S_\phi]$ in a certain neighborhood[8] of i. Set $S_{\phi+1}$ is an independent set of $\tilde{\tilde{H}}_p^\mu[S_\phi]$ and is likely to be an MIS with respect to a certain neighborhood[8] of i. Sections 9.3.1 and 9.3.2 describe in detail how graphs $\tilde{\tilde{H}}_p^\mu[S_\phi]$ and sets $S_{\phi+1}$ are computed. While performing this computation, in each phase ϕ each node in S_ϕ transmits its respective bcast-message m for $\mathcal{O}(Q \cdot \log(1/\varepsilon_{approg}))$ time steps, in each time step with probability p/Q with $Q = \Theta(\log^\alpha \Lambda)$, see Lines 10–13. Denote by m' the first bcast-message transmitted during Line 11 that node i receives during an epoch. In Line 18 node i outputs $rcv(m')_i$.

9.3.1 Computation of Graph $\tilde{\tilde{H}}_p^\mu[S_\phi]$ and Schedule τ_ϕBased on S_ϕ in Line 8

We modify an algorithm described in [14] to do this. In this algorithm we change the number of times T that each message is sent. We define

$$T := \Theta\left(\frac{\log\left(\frac{f(h_1)}{\varepsilon_{approg}}\right)}{\gamma^2\mu}\right)$$

where h_1 is defined in Definition 9.2 and f is the function that bounds the growth of G, see Definition 4.1.

Definition 9.2. *For $\Phi = \Theta(\log \Lambda)$, we set $h_\Phi := h'_\Phi := 1$ and define recursively $h'_\phi := 3h_{\phi+1}$ and $h_\phi := h'_\phi + c\log^*(\Lambda/\varepsilon_{approg}) + 1$ for $1 \leq \phi < \Phi$, where c is chosen such that $c\log^*(\Lambda/\varepsilon_{approg})$ bounds the runtime of the MIS algorithm [47] when applied on a network with node-IDs $\in [1, \frac{poly\,\Lambda}{\varepsilon_{approg}}]$.*

We restate the algorithm of [14] with our modified parameter T in order to perform our localized analysis later. All nodes in S_ϕ transmit their ID for T rounds with probability p in each round. Each node maintains a list of IDs that it received and counts how often each ID was received. Each ID that was received at least $(1-\gamma/2)\mu T$ times is a potential $\tilde{\tilde{H}}_p^\mu[S_\phi]$-neighbor. In another T time slots, in each slot every node transmits all IDs of these $\mathcal{O}(1)$ potential neighbors[9], again with probability p in each slot. A node $u \in S_\phi$ considers node $v \in S_\phi$ to be a $\tilde{\tilde{H}}_p^\mu[S_\phi]$-neighbor if v is a potential neighbor of u and u appears in the list of potential neighbors of v that u received.

Schedule τ_ϕ keeps track of the nodes random choices to send depending on the time slot. That is τ_ϕ maps time slot $t \in \{1,\ldots,T\}$ to $\tau_\phi[t] \subseteq V$ of nodes that are sending in slot t.

[7]These graphs were abbreviated by H_ϕ in the high-level description in Section 9. We want to stress that the sets $S_1 \supseteq S_2 \supseteq \cdots \supseteq S_\Phi$ computed by our algorithm are likely to differ from the sets $S_1 \supseteq S_2 \supseteq \cdots \supseteq S_\Phi$ computed in [14], but might be the same with a very low probability.

[8]The size of this neighborhood is specified later in the analysis and not relevant in the specification of the algorithm, see Definition 10.5.

[9] Each node has at most $\frac{1}{(1-\gamma/2)\mu} = \mathcal{O}(1)$ many potential neighbors (as remarked in [14]).

9.3.2 Computation of Set $S_{\phi+1}$ Based on $\tilde{\tilde{H}}_p^\mu[S_\phi]$ and Schedule τ_ϕ in Line 9

The authors of [14] show how to simulate the MIS-algorithm of [47] on $\tilde{\tilde{H}}_p^\mu[S_\phi]$ and define $S_{\phi+1}$ to be the computed MIS. In order to perform a more localized analysis, we need to modify their approach. In particular the runtime of the deterministic MIS-algorithm [47] depends on the range from which node IDs are chosen, not on the network size. While [14] uses unique IDs $\in [1, \text{poly } n]$, which results in a runtime of $\mathcal{O}(\log^* n)$, we desire a runtime that depends only on local parameters.

In order to achieve such a runtime, we let each node $v \in S_\phi$ choose a temporary label $l_{i,\phi} \in \left[1, \frac{\text{poly } \Lambda}{\varepsilon_{approg}}\right]$ uniformly at random in each phase. Then we execute a modified version of the MIS-algorithm of [47] for the CONGEST model using these temporary labels. As these labels might not be unique, we need to modify the algorithm of [47], as it might not terminate when non-unique labels are used.

To state our modifications and being able to argue that this achieves the desired outcome, we review the algorithm of [47]. After this, we present our modification of it in the CONGEST model. Subsequently we adapt the simulation of CONGEST algorithms in this probabilistic graph/SINR model given in [14] to our modified parameters.

The MIS-algorithm of [47]: Each node starts in state *competitor* and can change its state during the computation between states $\{competitor, ruler, ruled, dominator, dominated\}$. At the end of the algorithm, the set of all nodes in state *dominator* is an MIS, and all other nodes are in state *dominated* – as shown in [47]. To achieve this, the network executes a number of stages until all nodes are in state *dominator* or *dominated*. At the beginning of each stage every node v that is in state *competitor* at that time sets a variable r_v to its ID. After this, the stage performs $\log^*(N) + 2$ phases, where N indicates the range $[1, N]$ from which IDs are chosen. In each phase a node v in state *competitor* 1) exchanges r_v with its neighbors, and 2) updates r_v as well as its state depending on r_v and the received $r_w, w \in N(v)$, from its neighbors. If in a phase $r_v < \min_{w \in N(v) \setminus \{v\}} r_w$ in that phase, node v changes its state to *dominator* and stays in that stage until the end of the algorithm. If $r_v = \min_{w \in N(v) \setminus \{v\}} r_w$, then v changes its state to *ruler*. If $r_v > \min_{w \in N(v) \setminus \{v\}} r_w$, then v updates r_v depending on the bit where r_v and $\min_{w \in N(v) \setminus \{v\}} r_w$ differ and might change its state to *dominated*/*ruled* in case a neighbor changed its state to *dominator*/*ruler*. The proof of [47] uses the fact that IDs in the network are unique to argue that after a constant number c' of stages all nodes are in state *dominator* or *dominated* and nodes only terminate once they reached one of these states.

Our modification of this algorithm in the CONGEST model: We modify this algorithm to set $r_v := l_{v,\phi}$ instead of using v's ID at the beginning of each stage. As temporary labels $l_{v,\phi}$ are not unique, it can happen that after c' many stages some nodes are neither in state *dominator* nor *dominated*. Therefore we change the algorithm to terminate at a predetermined time (after c' stages) instead of terminating at each node once it is in state *dominator* or *dominated*. We still choose $S_{\phi+1}$ to consist only of nodes in state *dominator* and ignore nodes not in state *dominator*/*dominated*.

Adapted simulation of CONGEST algorithms in our probabilistic graph/SINR model: Similar to [14], each round of communication in the CONGEST model is simulated by T time steps in our model, where we use T as defined above. In each time step $t \in \{1, \ldots, T\}$ the messages (sent in a round of the algorithm for the CONGEST model) is sent by nodes $\tau_\phi[t]$, such that no messages are unsuccessful.

In contrast to [14], our analysis requires that nodes know if their messages arrived at the destination. Such an acknowledgment can be implemented as node i knows from which neighbors in $\tilde{\tilde{H}}_p^\mu[S_\phi]$ it should receive a message within time T (as we just computed $\tilde{\tilde{H}}_p^\mu[S_\phi]$). We can acknowledge received messages by splitting each time slot into two slots, a transmission and an acknowledgment slot. This implies that the (reliability) probability of an acknowledged transmission is μ^2. While w.h.p. communication is reliable in [14], it turns out that we cannot make these guarantees due to our choice of T. Therefore we say that communication at node $u \in S_\phi$ was unsuccessful (in phase ϕ) when node u did not receive messages (and acknowledgments for reception of its own messages) from all its S_ϕ-neighbors within time T. Once communication was unsuccessful, a node $u \in S_\phi$ stops participating in this epoch and does not join $S_{\phi+1}$ in this epoch. A node $u \in S_\phi$ that stopped during the current epoch starts participating again in the next epoch as long as it has an ongoing broadcast. Messages received from nodes that are not $\tilde{\tilde{H}}_p^\mu[S_\phi]$-neighbors are ignored and not acknowledged.

10 Analysis of our Implementation of Approximate Progress

We start with an outline of the analysis in Section 10.1 for the implementation of approximate progress of Section 9. This is followed by Sections focusing on details of different issues mentioned in that outline.

10.1 Outline of the Analysis

We analyze the effect of the two main modifications of the algorithm of [14] with respect to their analysis and put it into the context of approximate progress. We outline the effects of these modifications here together with our approach before we dive into details in the next sections.

10.1.1 First Modification: Non-Unique Labels in the MIS Computation

This difference is rooted in our modification of the MIS-algorithm of [47] combined with using non-unique temporary labels $\in [1, \frac{\text{poly}\,\Lambda}{\varepsilon_{approg}}]$ instead of unique IDs $\in [1, \text{poly}\, n]$ in [14]. In Section 10.2 (Lemma 10.1) we argue in the model of [47] the sets S_ϕ computed by our modified MIS-algorithm are independent sets in $\tilde{H}_{\phi-1}$. Furthermore, for any given node v, with probability $1 - \varepsilon_{approg}/3$, this set is maximal in a neighborhood around v "large enough" to ensure that this part of computations involved in approximate progress at node v is correct.

10.1.2 Second Modification: Fewer Repetitions of Transmissions

In the algorithm of [14] each node sends every bcast-message $\mathcal{O}(\log^\alpha(\Lambda)\log n)$ times, while we use only $\mathcal{O}(\log^\alpha(\Lambda)\log(1/\varepsilon_{approg}))$ repeated transmissions. This implies that [14] can assume that all communication is successful at any point w.h.p.. For large ε_{approg} we only have weak probability guarantees for success of communication. One side-effect is that with very high probability the computed graphs $\tilde{\tilde{H}}_\phi$ are not the desired global approximations of graphs H_ϕ. This in turn affects correctness of approximate progress and we need to analyze local and global implications caused by reducing the number of repeated transmissions.

1. **Global implications of unsuccessful transmissions:** Global interference might increase in the long term and we need to bound this. Unsuccessful transmissions that are undetectable as the receiver does not know from which other nodes to expect messages can only appear 1) during the computation of $\tilde{\tilde{H}}^\mu_p[S_\phi]$, and 2) while transmitting the message in Line 11. The latter will not cause increased global interference in the long term, as it does not influence the activity of nodes in future phases of the current epoch. Thus we only need to consider unsuccessful transmissions during the computation of $\tilde{\tilde{H}}^\mu_p[S_\phi]$. Consider a node v and assume node v has computed a wrong set of neighbors, that is a set of neighbors that does not correspond to a $(1-\gamma)$-approximation of $H^\mu_p[S_\phi]$. In such a case we just assume for the sake of worst case analysis of additional interference that v joins the MIS $S_{\phi+1}$ of $\tilde{\tilde{H}}^\mu_p[S_\phi]$ – regardless of where v actually joins or not. Denote the set of all these nodes with wrong neighborhoods that unconsciously might cause additional interference during the current epoch by W. (Note that at the beginning of each epoch $W = \emptyset$, as no unsuccessful transmission happened yet.) We bound the additional (global) interference caused in case all nodes in W erroneously decided to join S_ϕ in Lemma 10.3 (regardless of which nodes in W actually join S_ϕ). Each time when we need to make an argument related to interference from nodes in S_ϕ in subsequent proofs, we also argue that the additional interference from nodes in W is negligible compared to interference from a correctly computed S_ϕ. Note that thus interference from nodes in W might be counted twice (in particular nodes in $S_\phi \cap W$), but this does not hurt the analysis. After $\tilde{\tilde{H}}_\phi$ is computed, all transmissions are successful. They use the same schedule used to compute $\tilde{\tilde{H}}_\phi$.
2. **Local implications of unsuccessful transmissions:** Local communication of messages, which is based on the success of repeated transmissions, must be successful in a certain area[8] around v to ensure that 1) a node v that has a broadcasting $G_{1-2\varepsilon}$ neighbor receives a bcast-message from a broadcasting $G_{1-\varepsilon}$-neighbor in case all local computations are correct, and 2) we can transfer and extend tools from [13] to our localized analysis. This area in which this needs to be true contains all nodes possibly involved in the selection of a node from which v might receive a bcast-message. These unsuccessful transmissions can only appear during the computation of $\tilde{\tilde{H}}^\mu_p[S_\phi]$ and while transmitting the bcast-message in Line 11. Only if communication is locally successful, it is guaranteed that graph $\tilde{\tilde{H}}^\mu_p[S_\phi]$ is an $(1-\gamma)$- approximation of $H^\mu_p[S_\phi]$ w.r.t. the above mentioned neighborhood of v, which is necessary in order to transfer the analysis of [14]. We analyze the probability that $\tilde{\tilde{H}}^\mu_p[S_\phi]$ is locally an approximation in Lemma 10.10. Finally, approximate progress is made only if communication of bcast-messages succeeds locally.
   For all $G_{1-\varepsilon}$-neighbors u' of v (from which v might receive a bcast-message), we lower bound the probability that all the above local computations/transmissions involved in the broadcast of u' are successful in Lemme 10.13.

10.2 Local Effects of Non-Unique Labels

We start by analyzing the effect of using (potentially) non-unique labels chosen uniformly at random $\in \left[1, \frac{\text{poly}\,\Lambda}{\varepsilon_{approg}}\right]$ in the modified MIS computation, which is the first difference to [14], as pointed out in Section 10.1.1.

Lemma 10.1. *Let $H = (V, E)$ be a constant degree growth-bounded graph and let $U \subseteq V$ be a set of nodes of size at most $\mathcal{O}(\Lambda^2)$. Consider an execution of our modification of the MIS-algorithm of [47] on H in the CONGEST model using random labels $\in \left[1, \frac{poly\,\Lambda}{\varepsilon_{approg}}\right]$. Then the set of nodes in state dominator is 1) an independent set, and 2) with probability at least $1-\frac{\varepsilon_{approg}}{3\Phi}$ this set is maximal with respect to $N_{H,c\cdot 4^{\Phi}\cdot\log^*(\Lambda/\varepsilon_{approg})}(U)$, the $c\cdot 4^{\Phi}\cdot\log^*(\Lambda/\varepsilon_{approg})$-neighborhood of U in H.*

Proof. From the description of the algorithm it follows that no neighboring nodes can be in state *dominator*. Therefore the set of *dominators* remains an independent set despite our modification.

Due to the analysis of [47], which uses that H is growth bounded, we know that in case of unique IDs the algorithm computes an MIS within $c' < c$ stages. As the runtime of the algorithm is $c\log^*(\Lambda/\varepsilon_{approg})$, only the $c\log^*(\Lambda/\varepsilon_{approg})$-neighborhood of $N_{H,c\cdot 4^{\Phi}\cdot\log^*(\Lambda/\varepsilon_{approg})+1}(U)$ is involved in deciding which nodes among $N_{H,c\cdot 4^{\Phi}\cdot\log^*(\Lambda/\varepsilon_{approg})+1}(U)$ change their state to *dominator*. From this we can conclude that if nodes in $N_{H,c\cdot 4^{\Phi}\cdot\log^*(\Lambda/\varepsilon_{approg})+1+c\log^*(\Lambda/\varepsilon_{approg})}(U)$ chose unique temporary labels, then the set of nodes in state *dominator* is a maximal independent set with respect to $N_{H,c\cdot 4^{\Phi}\cdot\log^*(\Lambda/\varepsilon_{approg})}(U)$. Note, that as we considered *dominators* located in $N_{H,c\cdot 4^{\Phi}\cdot\log^*(\Lambda/\varepsilon_{approg})+1}(U)$ for maximality in $N_{H,c\cdot 4^{\Phi}\cdot\log^*(\Lambda/\varepsilon_{approg})}(U)$, it cannot happen that there is a node at the border of $N_{H,c\cdot 4^{\Phi}\cdot\log^*(\Lambda/\varepsilon_{approg})}(U)$ that has no neighbor in state *dominator*.

Now observe that it is $c\cdot 4^{\Phi}\cdot\log^*(\Lambda/\varepsilon_{approg})+c\log^*(\Lambda/\varepsilon_{approg}) = c\text{poly }(\Lambda)\cdot\log^*(\Lambda/\varepsilon_{approg})$, as $\Phi = \Theta(\log\Lambda)$. As H is growth bounded and has constant degree, this implies that there are at most $|U|\cdot\text{poly }\Lambda = \text{poly }\Lambda$ nodes involved in the state-changes of nodes in $N_{H,4^{\Phi}\cdot\log^*(\Lambda/\varepsilon_{approg})+1}(U)$. As we choose temporary labels from $\left[1, \frac{\text{poly}\,\Lambda}{\varepsilon_{approg}}\right]$, we can choose this range large enough such that with probability at least $1-\frac{\varepsilon_{approg}}{3\Phi}$ the labels are unique among the poly Λ nodes in the $c\cdot 4^{\Phi}\cdot\log^*(\Lambda/\varepsilon_{approg})+c\log^*(\Lambda/\varepsilon_{approg})$-neighborhood of U. □

10.3 Global Effects of Unsuccessful Transmissions

We analyze Case 1.a pointed out in Section 10.1.2, i.e. we bound the global interference from nodes with undetectable unsuccessful transmissions.

Definition 10.2 (Set W of nodes with wrong neighborhoods (due to unsuccessful transmissions))**.** *Denote by $W \subseteq S_1$ the set of all those nodes v such that for at least one $\phi \in \{1, \cdots, \Phi\}$ it is not the case that $N_{H_p^{\mu}[S_\phi]}(v) \subseteq N_{\tilde{H}_p^{\mu}[S_\phi]}(v) \subseteq N_{H_p^{\mu(1-\gamma)}[S_\phi]}(v)$, i.e. v's direct neighborhood does not $(1-\gamma)$-approximate $N_{H_p^{\mu}[S_\phi]}(v)$.*

Lemma 10.3. *Given point i in space, the expected total additional interference $I_W(i)$ that point i receives from all nodes in W at any given time is less than $(\frac{\varepsilon_{approg}}{\Lambda})^{\Theta(1)}$.*

Proof. We first bound the probability that $N_{\tilde{H}_p^{\mu}[S_\phi]}(v)$ does not correspond to a $(1-\gamma)$-approximation of $H_p^{\mu}[S_\phi]$ during a single phase. Let's consider a potential edge $(u, v) \in S_\phi \times S_\phi$.

As each ID is transmitted T times, a Chernoff bound implies that an edge (u,v) is included in $\tilde{\tilde{H}}_p^\mu[S_\phi]$ if and only if (u,v) belongs to a $(1-\gamma)$-approximation of $H_p^\mu[S_\phi]$ with probability at least

$$1 - e^{-\Theta(T)} = 1 - e^{-\Theta\left(\log \frac{f(h_1)}{\varepsilon_{approg}}\right)} \geq 1 - \left(\frac{\varepsilon_{approg}}{f(h_1)}\right)^{\Theta(1)}. \tag{2}$$

The constant hidden in the Θ-notation depends on μ, γ and the Chernoff bound used. Recall[9] that node v has constant many neighbors in $H_p^\mu[S_\phi](v)$. Therefore the probability that $N_{\tilde{\tilde{H}}_p^\mu[S_\phi]}(v)$ is a $(1-\gamma)$-approximation of $N_{H_p^\mu[S_\phi]}(v)$ is at least $\left(1 - \left(\frac{\varepsilon_{approg}}{f(h_1)}\right)^{\Theta(1)}\right)^{\Theta(1)} \leq 1 - \left(\frac{\varepsilon_{approg}}{f(h_1)}\right)^{\Theta(1)}$.

From this we conclude, that in each square of size $R_{1-2\varepsilon}$ times $R_{1-2\varepsilon}$ the expected number of nodes that incorrectly have no edges in at least one of the Φ phases is at most

$$\left(\frac{\varepsilon_{approg}}{\Lambda}\right)^{\Theta(1)} \cdot \Lambda^2 \cdot \Phi = \left(\frac{\varepsilon_{approg}}{\Lambda}\right)^{\Theta(1)}.$$

Now assume that exactly the nodes in W transmit at the same time. We use a standard argument from the SINR community to bound the expected interference that node i receives from nodes in W similar to the one in [24]. For the analysis we assume that the plane is partitioned into a $R_{1-2\varepsilon}$-grid centered in v. Denote by A_d the set of grid-cells that contain nodes of L_0-distance at least $(d-1)\cdot R_{1-2\varepsilon}$ and at most $d \cdot R_{1-2\varepsilon}$ to i. Therefore A_d contains $8d-4$ squares of size $R_{1-2\varepsilon}$ times $R_{1-2\varepsilon}$.

From this we conclude that the expected number of nodes in $A_d \cap W$ is upper bounded by $\mathcal{O}\left(\left(\frac{\varepsilon_{approg}}{\Lambda}\right)^{\Theta(1)} \cdot d\right)$.

Each node in A_d is at Euclidean distance at least $d-1$ to i, such that the interference caused at i by a single node in A_d sending with power P is at most $P/(d-1)^\alpha$. Therefore the expected interference at point i from nodes in $A_d \cap W$ is upper bounded by $\mathcal{O}\left(\left(\frac{\varepsilon_{approg}}{\Lambda}\right)^{\Theta(1)} / d^{\alpha-1}\right)$, where we use that power P is constant. Now we can upper bound the expected interference that point i receives from W by

$$\begin{aligned} I_W(i) &= \sum_{d=1}^{\infty} I_{A_d \cap W}(i) = \sum_{d=1}^{\infty} \mathcal{O}\left(\left(\frac{\varepsilon_{approg}}{\Lambda}\right)^{\Theta(1)} / d^{\alpha-1}\right) \\ &= \mathcal{O}\left(\left(\frac{\varepsilon_{approg}}{\Lambda}\right)^{\Theta(1)}\right) = \left(\frac{\varepsilon_{approg}}{\Lambda}\right)^{\Theta(1)}, \end{aligned}$$

where we use $\alpha > 2$ in the second-last bound and the fact that p-series with $p > 1$ converge to a constant.

Finally, note that each node actually sends only with probability p (or p/Q) during the execution of each phase. Replacing the assumption that all node in W transmit at the same time by these probabilities implies that the expected interference remains $\left(\frac{\varepsilon_{approg}}{\Lambda}\right)^{\Theta(1)}$, as p is constant and $Q = \mathcal{O}(\log^\alpha(\Lambda))$. $\square$

10.4 Local Effects of Unsuccessful Transmission

We analyze Case 2 pointed out in Section 10.1.2. We start with a bound on h_1 (see Definition 9.2), define local success of an epoch (see Definition 10.8) and then analyze the probability of local success of an epoch. Lemma 10.12 is the main Lemma of this section and states that for any set $S_1 \subseteq V$ and node $i \in N_{G_{1-2\varepsilon}}(S_1)$ two out of three properties of a successful epoch are satisfied with probability at least $1 - \varepsilon_{approg}/3$ at point i.

Lemma 10.4. *The following is true: $3^{\Phi-1} \leq h_1 \leq c \cdot 4^{\Phi} \cdot \log^*(\Lambda/\varepsilon_{approg})$ for all parameters $\Phi, \Lambda, \varepsilon_{approg}$ in the ranges considered in this paper.*

Proof. The first bound is immediate. To derive the second bound we show by induction on ϕ that $h_\phi \leq c \cdot 4^{\Phi-\phi} \cdot \log^*(\Lambda/\varepsilon_{approg})$. It is $h_\Phi = 1 \leq c \log^*(\Lambda/\varepsilon_{approg})$. For $\phi \leq \Phi$ assume that $h_\phi = c \cdot 4^{\Phi-\phi} \cdot \log^*(\Lambda/\varepsilon_{approg})$, then it is

$$\begin{aligned} h_{\phi-1} &= c \cdot 4^{\Phi-\phi} \cdot \log^*(\Lambda/\varepsilon_{approg}) \cdot 3 + c \log^*(\Lambda/\varepsilon_{approg}) \\ &= c \cdot 4^{\Phi-\phi} \cdot \log^*(\Lambda/\varepsilon_{approg}) \cdot 3 + c \cdot 4^{\Phi-\phi} \cdot \log^*(\Lambda/\varepsilon_{approg}) \\ &= c \cdot 4^{\Phi-(\phi-1)} \cdot \log^*(\Lambda/\varepsilon_{approg}) \end{aligned}$$

□

10.4.1 Definition of Local Success of an Epoch

Given point i, we define the sets of nodes involved in the local computation that selects a node from which i might receive a bcast-messages in phase ϕ.

Definition 10.5 (Sets $U_{\phi,i}, S_{\phi,i}$ and $S'_{\phi,i}$). *Let $i \in N_{G_{1-2\varepsilon}}(S_1)$ be a node (of which we can think as a point in space) that has a $G_{1-2\varepsilon}$-neighbor with an ongoing broadcast. Let $U_{\phi,i} := N_{G_{1-\varepsilon}}(i) \cap S_\phi$ be the subset of nodes at distance at most $R_{1-2\varepsilon}$ from which i might receive a bcast-message in phase ϕ. We define sets $S_{\phi,i}$ and $S'_{\phi,i}$:*

- *$S_{\phi,i} := N_{\tilde{\tilde{H}}^\mu_p[S_\phi],h_\phi}(U_{\phi,i})$, the h_ϕ-hop $\tilde{\tilde{H}}^\mu_p[S_\phi]$-neighborhood of U_ϕ.*
- *$S'_{\phi,i} := N_{\tilde{\tilde{H}}^\mu_p[S_\phi],h'_\phi}(U_{\phi,i}) \subseteq S_{\phi,i}$, the h'_ϕ-hop $\tilde{\tilde{H}}^\mu_p[S_\phi]$-neighborhood of $U_{\phi,i}$.*

Definition 10.6. *(Local success of computing $\tilde{\tilde{H}}^\mu_p[S_\phi]$ and $S_{\phi+1}$).*

A computation of $\tilde{\tilde{H}}^\mu_p[S_\phi]$ is successful at point i if $\tilde{\tilde{H}}^\mu_p[S_\phi]|_{S_{\phi,i}}$ corresponds to a $(1-\gamma)$-approximation of $H^\mu_p[S_\phi]|_{S_{\phi,i}}$. A computation of independent set $S_{\phi+1}$ on $\tilde{\tilde{H}}^\mu_p[S_\phi]$ is successful at node i if $S_{\phi+1}$ is a (ϕ, i)-locally maximal independent set in the sense that:

1. *$S_{\phi+1}$ is independent in $\tilde{\tilde{H}}^\mu_p[S_\phi]$, and*
2. *there is no node $v \in S_\phi \setminus S_{\phi+1,i}$ such that $S_{\phi+1,i} \cup \{v\}$ is independent in $\tilde{\tilde{H}}^\mu_p[S_\phi]$ and v is of distance at most h_ϕ to any $u \in U_{\phi+1,i}$ with respect to $\tilde{\tilde{H}}^\mu_p[S_{\phi+1} \cup \{v\}]$.*

Adding a single node v to the vertex-set $S_{\phi+1}$ might change the topology of $\tilde{\tilde{H}}^\mu_p[S_{\phi+1}]$, and thus distances in other parts of the graph due to SINR constraints. Therefore we need to show that this definition of (ϕ, i)-local maximality is well-defined.

Lemma 10.7. *The definition of (ϕ, i)-local maximality is well-defined, i.e. (ϕ, i)-local maximality of set $S_{\phi+1}$ is invariant to adding a node v that is independent to $S_{\phi+1}$ in $\tilde{\tilde{H}}_p^\mu[S_\phi]$.*

Proof. For any $u_1, u_2 \in S_{\phi+1}$ the distance between u_1 and u_2 in $\tilde{\tilde{H}}_p^\mu[S_{\phi+1} \cup \{v\}]|_{S_{\phi+1,i}}$ compared to the distance in $\tilde{\tilde{H}}_p^\mu[S_{\phi+1}]|_{S_{\phi+1,i}}$ might potentially

1. decrease, as there might now be a shorter u_1, u_2-path via v, or
2. increase, as v adds interference, which might reduce connectivity among the nodes in $S_{\phi+1}$. Even though there might now be a short-cut for part of the paths via v, added interference might still cause the overall path to be longer.

Therefore, in case that v is not at distance at most $h_{\phi+1}$ to any node in a set $S_{\phi+1,i}$, set $S_{\phi+1,i}$ stays (ϕ, i)-locally maximal independent of adding v, as the only way a node can be closer to i is via a path through v—and v in turn is at distance at least $h_{\phi+1}$. □

Definition 10.8 (Local success of an epoch)**.** *An epoch is successful at point i if*

1. *the computations of each graph $\tilde{\tilde{H}}_p^\mu[S_1], \dots, \tilde{\tilde{H}}_p^\mu[S_\Phi]$ are successful at point i, and*
2. *the computations of each set $S_2, \dots, S_\Phi$ are successful at point i, and*
3. *there is a $\phi \in \{1, \dots, \Phi\}$, such that i receives the bcast-message m transmitted by some node $u_\phi \in U_{\phi,i}$ in Line 11 of phase ϕ.*

Note that in the proofs of this Section we never assume that we know the location of i nor that we know u_ϕ or u_ϕ's location/distance to i.

10.4.2 Probability of Local Success of Computing Graph $\tilde{\tilde{H}}_p^\mu[S_\phi]$ Based on S_ϕ

This is an important step towards analyzing Property 1 of local success of an epoch. Using this property we later iteratively guarantee local success of computing graphs $\tilde{\tilde{H}}_p^\mu[S_\phi]$ in each phase. As the nodes involved in decisions of other nodes cannot be too far away, this helps to bound the probability that locally correct computations take place. If all involved computations at nodes in transmission range are successful, approximate progress takes place.

Remark 10.9. *In the remaining part of Section 10.4 we focus only on unsuccessful transmission to keep the analysis clean. Therefore we assume for now that $S_{\phi,i}$ is assigned unique temporary labels such that in absence of unsuccessful transmissions the modified MIS-algorithm always computes a set that is $(\phi + 1, i)$-locally maximal. In Section 10.7 we argue that this assumption can be dropped at the cost of the probability derived in Lemma 10.1.*

Lemma 10.10. *Consider a node $i \in S_1$ and phase ϕ of Algorithm 9.1. Line 8 described in Section 9.3.1 computes a graph $\tilde{\tilde{H}}_p^\mu[S_\phi]$ in time $\mathcal{O}(\Phi + \log(1/\varepsilon_{approg}))$, such that with probability at least $1 - \left(\frac{\varepsilon_{approg}}{f(h_1)}\right)^{\Theta(1)}$ the computation of graph $\tilde{\tilde{H}}_p^\mu[S_\phi]$ is successful at node i. Given set S_ϕ, the decision whether an edge (u, v) is in graph $\tilde{\tilde{H}}_p^\mu[S_\phi]$ does not involve communication between nodes other than u and v.*

Proof. From Bound 2 in Lemma 10.3 we know that the probability that an edge $(u, v) \in S_\phi \times S_\phi$ that belongs to a $(1-\gamma)$-approximation of $H_p^\mu[S_\phi]$ is included in $\tilde{\tilde{H}}_p^\mu[S_\phi]$, h_ϕ is at least $1 - \left(\frac{\varepsilon_{approg}}{f(h_1)}\right)^{\Theta(1)}$. As

- $S_{\phi,i}$ is defined using the h_ϕ-hop $\tilde{\tilde{H}}_p^\mu[S_\phi]$-neighborhood of nodes $U_{\phi,i}$ (see Definition 10.5), and
- $U_{\phi,i}$ contains at most Λ^2 many edges (see Definition 10.5), and
- $\tilde{\tilde{H}}_p^\mu[S_\phi]$ has degree[9] $\mathcal{O}(1)$, and $\tilde{\tilde{H}}_p^\mu[S_\phi]$ is growth bounded by f,

there are at most $\mathcal{O}(f(h_\phi) \cdot \Lambda^2)$ edges in $\tilde{\tilde{H}}_p^\mu[S_\phi]|_{S_{\phi,i}}$ among which the algorithm needs to choose $\tilde{\tilde{E}}_p^\mu[S_\phi] \cap (S_{\phi,i} \times S_{\phi,i})$ correctly. Therefore the probability that $\tilde{\tilde{E}}_p^\mu[S_\phi] \cap (S_{\phi,i} \times S_{\phi,i})$ of edges among nodes $S_{\phi,i}$ is chosen in a way that $(1-\gamma)$-approximates edges in $H_p^\mu[S_\phi]|_{S_{\phi,i}}$ is at least

$$\left(1 - \left(\frac{\varepsilon_{approg}}{f(h_1)}\right)^{\Theta(1)}\right)^{\mathcal{O}\left(f(h_\phi)\Lambda^2\right)} \geq 1 - \mathcal{O}(f(h_\phi)\Lambda^2) \cdot \left(\frac{\varepsilon_{approg}}{f(h_1)}\right)^{\Theta(1)} = 1 - \left(\frac{\varepsilon_{approg}}{f(h_1)}\right)^{\Theta(1)},$$

where we use

- $h_\phi \leq \operatorname{poly} \Lambda$, as $h_\phi \leq h_1$, $h_1 \geq 3^{\Phi-1} \geq \Phi$ (see Lemma 10.4), and $\Phi := \Theta(\log \Lambda)$; and
- that the growth bound f is a monotonic increasing function (as the number of neighbors can only grow with the distance), and
- that we can choose the constant hidden in the Θ-notation arbitrarily high.

Furthermore, as μ and γ are constants, and as $h_1 \leq c \cdot 4^\Phi \cdot \log^*(\Lambda/\varepsilon_{approg})$ (see Lemma 10.4), and as f is a polynomial function, we can bound the runtime T by

$$\begin{aligned} T &= \Theta\left(\frac{\log \frac{f(h_1)}{\varepsilon_{approg}}}{\gamma^2 \mu}\right) = \Theta\left(\log \frac{f\left(c \cdot 4^\Phi \cdot \log^*(\Lambda/\varepsilon_{approg})\right)}{\varepsilon_{approg}}\right) \\ &= \Theta\left(\Phi + \log(\log^*(\Lambda/\varepsilon_{approg})) + \log(1/\varepsilon_{approg})\right) = \Theta\left(\Phi + \log(1/\varepsilon_{approg})\right), \end{aligned}$$

where we choose the constant hidden in the Θ-notation sufficiently high. Finally, note that in this process the decision whether an edge (u, v) is in the graph $\tilde{\tilde{H}}_p^\mu[S_\phi]$ does not involve communication between nodes other than u and v. □

10.4.3 Probability of Local Success of Computing Set $S_{\phi+1}$ Based on $\tilde{\tilde{H}}_p^\mu[S_\phi]$

This is an important step towards the analysis of Property 2 of local success of an epoch. Using this property we later iteratively guarantee local success of computing sets S_ϕ in each phase. As the nodes involved in decisions of other nodes cannot be too far away, this helps to bound the probability that locally correct computations take place. If all involved computations at nodes in transmission range are successful, approximate progress takes place.

Lemma 10.11. *Given graph $\tilde{\tilde{H}}_p^\mu[S_\phi]$, consider phase ϕ in Algorithm 9.1. Line 9 described in Section 9.3.2 computes in time $\mathcal{O}\left((\Phi + \log(1/\varepsilon_{approg}))\log^*(\Lambda/\varepsilon_{approg})\right)$ a set $S_{\phi+1}$ that is an independent set in $\tilde{\tilde{H}}_p^\mu[S_\phi]$. The computation of set $S_{\phi+1,i}$ is successful at point i. Furthermore, determining the $S_{\phi+1,i}$ part of $S_{\phi+1}$ involves only nodes in $S_{\phi,i}$.*

Proof. **Runtime analysis:** The algorithm described in Section 9.3.2 consists of simulating (in the SINR model) an algorithm to compute an MIS in the CONGEST model taking $c\log^*(\Lambda/\varepsilon_{approg})$ rounds in the CONGEST model. The provided simulation of each round of the CONGEST model in the SINR model takes $\mathcal{O}(T)$ time slots. Therefore the total runtime is

$$\begin{aligned}
\mathcal{O}(T \cdot \log^*(\Lambda/\varepsilon_{approg})) &= \mathcal{O}\left(\log\left(\frac{f(h_1)}{\varepsilon_{approg}}\right) \cdot \log^*(\Lambda/\varepsilon_{approg})\right) \\
&= \mathcal{O}\left(\log\left(\frac{f\left(c \cdot 4^\Phi \cdot \log^*(\Lambda/\varepsilon_{approg})\right)}{\varepsilon_{approg}}\right) \cdot \log^*(\Lambda/\varepsilon_{approg})\right) \\
&= \mathcal{O}\left((\Phi + \log(\log^*(\Lambda/\varepsilon_{approg})) + \log(1/\varepsilon_{approg})) \cdot \log^*(\Lambda/\varepsilon_{approg})\right). \\
&= \mathcal{O}\left((\Phi + \log(1/\varepsilon_{approg})) \cdot \log^*(\Lambda/\varepsilon_{approg})\right),
\end{aligned}$$

where we use similar arguments as in the runtime analysis of Lemma 10.10 as well as the fact that $\log\log^*(\Lambda) \le \Theta(\log(\Lambda)) = \Phi$.

An independent set is computed: This algorithm simulates the MIS algorithm of [47], which computes an MIS in growth-bounded graphs, and attempts to compute a subset $S_{\phi+1}$ of an MIS on $\tilde{\tilde{H}}_p^\mu[S_\phi]$. The algorithm might not achieve maximality as nodes might stop participating in this epoch after their communication was unsuccessful[10]. As these nodes do not join $S_{\phi+1}$, set $S_{\phi+1}$ is still an independent set in $\tilde{\tilde{H}}_p^\mu[S_\phi]$.

Given graph $\tilde{\tilde{H}}_p^\mu[S_\phi]$, the computation of set $S_{\phi+1}$ is successful at point i : As all communication in $\tilde{\tilde{H}}_p^\mu[S_\phi]|_{S_{\phi,i}}$ is successful due to using the same schedule τ_ϕ as when computing $\tilde{\tilde{H}}_p^\mu[S_\phi]$, the set $S_{\phi+1} \cap S'_{\phi,i}$ is (ϕ, i)-locally maximal. Recall that set $S'_{\phi,i}$ depends on $h'_\phi := 3h_{\phi+1}$, a choice taking into account that each hop in $\tilde{\tilde{H}}_p^\mu[S_{\phi+1}]|_{S_{\phi+1,i}}$ corresponds to at most 3 hops in $\tilde{\tilde{H}}_p^\mu[S_\phi]|_{N_{\tilde{\tilde{H}}_p^\mu[S_\phi]}(S_{\phi+1,i})}$, as otherwise $(\phi+1, i)$-local maximality of $S_{\phi+1}$ in $\tilde{\tilde{H}}_p^\mu[S_\phi]|_{N_{\tilde{\tilde{H}}_p^\mu[S_\phi]}(S_{\phi+1,i})}$ was violated. Therefore we conclude that $N_{\tilde{\tilde{H}}_p^\mu[S_\phi]}(S_{\phi+1,i}) \subseteq S'_{\phi,i}$. Furthermore any node v that could be added to $S_{\phi+1,i}$ without violating independence in $\tilde{\tilde{H}}_p^\mu[S_\phi]$ is at least $h'_\phi + 1$ hops away from $u_{\phi+1}$ in $\tilde{\tilde{H}}_p^\mu[S_\phi]$ and thus at least $h_{\phi+1} + 2$ hops away from u_ϕ in $\tilde{\tilde{H}}_p^\mu[S_{\phi+1}]$. Therefore Definition 10.8 of local successful computation of $S_{\phi+1,i}$ is satisfied.

Also note that only nodes in $N_{\tilde{\tilde{H}}_p^\mu[S_\phi], c\log^*(\Lambda/\varepsilon_{approg})}(S'_{\phi,i}) = S_{\phi,i}$ are involved in the computation, as the runtime of the MIS algorithm of [47] is $c\log^*(\Lambda/\varepsilon_{approg})$. □

[10] Recall that communication at node $u \in S_\phi$ is unsuccessful if u did not receive a message (and acknowledgments) for own messages form each neighbor in $\tilde{\tilde{H}}_p^\mu[S_\phi]$.

10.4.4 Probability of Satisfying Properties 1 and 2 of a Local Successful Epoch

Lemma 10.12. *For any set $S_1 \subseteq V$ and node $i \in N_{G_{1-2\varepsilon}}(S_1)$, both Properties 1 and 2 of Definition 10.8 of a successful epoch at point i are satisfied with probability at least $1-\varepsilon_{approg}/3$.*

Proof. Due to Lemma 10.10 only nodes in $S_{\phi,i}$ are involved in computing $\tilde{\tilde{H}}_p^\mu[S_\phi]|_{\phi,i}$ and Lemma 10.11 states that only this part of the graph is involved in computing $S_{\phi+1,i}$. By induction we only need to bound the probability that all graphs $\tilde{\tilde{H}}_p^\mu[S_1], \dots, \tilde{\tilde{H}}_p^\mu[S_{\Phi-1}]$ and all sets $S_2, \dots, S_\Phi$ are computed successfully at point i to prove the statement.

Due to Lemma 10.10 the probability that any of the graphs $\tilde{\tilde{H}}_p^\mu[S_1], \dots, \tilde{\tilde{H}}_p^\mu[S_{\Phi-1}]$ is computed successfully at point i is at least $1 - \left(\frac{\varepsilon_{approg}}{f(h_1)}\right)^{\Theta(1)}$. The probability that all of the sets $S_2, \dots, S_\Phi$ are computed successfully at point i is 1 due to Lemma 10.11. Notice that $1 - \left(\frac{\varepsilon_{approg}}{f(h_1)}\right)^{\Theta(1)}$ can be lower bounded by $(1 - \frac{\varepsilon_{approg}}{3^{\Phi-1}})^{\Theta(1)}$, as $h_1 \geq 3^{\Phi-1}$ (Lemma 10.4) and f is a monotonic increasing polynomial. While we obtain this generous bound as a side effect of other parts of the analysis, it is sufficient for our purposes to use $1 - \frac{\varepsilon_{approg}}{6\Phi}$ as a lower bound for this probability. Here, we assume $\Phi = \Theta(\log \Lambda) \geq 4$ for simplicity of the presentation. As there are Φ phases, in total Φ graphs need to be computed. Thus the probability that all these computations are successful at point i is at least $(1 - \frac{\varepsilon_{approg}}{6\Phi})^\Phi \geq 1 - \varepsilon_{approg}/3$. □

10.5 Probability of Approximate Progress Conditioned on Satisfaction of Property 3 of a Local Successful Epoch

After proving initial lemmas in the previous subsections, we first give an outline how these connect to the remaining parts of the proof of Theorem 9.1 via this section. In Lemma 10.12 we argued that with probability at least $1 - \varepsilon_{approg}/3$ we can assume that Properties 1 and 2 of Definition 10.8 of a successful epoch at point i are satisfied. Therefore we assume in Lemma 10.13 that Properties 1 and 2 of Definition 10.8 of a successful epoch at point i are satisfied, and show that in this case there is a phase $\phi' \in \{1, \dots, \Phi\}$ such that i could be able to receive a bcast-message m from a $G_{1-\varepsilon}$-neighbor. Node i will receive such a message if Property 3 of Definition 10.8 of a successful epoch at point i is satisfied. As we cannot yet analyze the probability of satisfaction of Property 3, we condition our probabilities in this section on satisfaction of Property 3. To analyze probability of satisfaction of Property 3, we first need to bound the runtime of an epoch, which is done in Lemma 10.18. In Section 10.7 we analyze the probability that i indeed receives m in phase ϕ' from a $G_{1-\varepsilon}$-neighbor by combining results from this and previous sections with a bound on the probability for satisfaction of Property 3 of Definition 10.8 of a successful epoch at point i. Section 10.7 also concludes the proof of the bound on f_{approg} stated in Theorem 9.1.

The main Lemma that we prove in this Section is

Lemma 10.13. *Given set S_1 and a node i and let there be a $G_{1-2\varepsilon}$-neighbor of i with an ongoing broadcast event $bcast(m)_j$. Assume Properties 1 and 2 of Definition 10.8 of a successful epoch at point i are satisfied. Then there is a phase $\phi' \in \{1, \dots, \Phi\}$ such that in phase ϕ node i receives a bcast-message from a $G_{1-\varepsilon}$-neighbor of i with probability $1 - \varepsilon_{approg}/3$.*

However, before we can proof Lemma 10.13, we need to derive a few more lemmas which are extended from [13] to our localized setting.

Lemma 10.14 (Extended version of Lemma 4.3 of [13]). *For all $p \in (0, 1/2]$, there is a $\mu \in (0, p)$ such that: Let $d_{\min} \leq R_{1-2\varepsilon}$ be the shortest distance between two nodes in a set $S \subseteq S_1$. Then the graph $H_p^{\mu}[S]$ contains all edges between pairs $u, v \in S$ for which $d(u, v) \leq \min\{2d_{\min}, R_{1-2\varepsilon}\}$.*

The part of the proof of Lemma 4.3 of [13] that changes relies on bounding the interference $I_S(u)$ that u receives from a set S. Compared to [13] we not only need to bound interference from a set S, but from $S \cup W$, as nodes in W might still participate in the computation and send messages due to unsuccessful transmissions in a previous phase that made them compute wrong neighborhoods. We show that one can choose slightly modified parameters in the algorithm/analysis such that the interference $I_{S\cup W}(u)$ is as small as in the original proof by [13]. Therefore other parts of their proof are not affected and can be immediately transferred.

For completeness we restate the full proof of [13] adapted to our modifications and extentions.

Proof. We restrict our attention to the case $d_{\min} < r_s/2$. If the minimum distance is between $r_s/2$ and r_s, the claim can be shown by a similar, simpler argument.

Consider some node $u \in S$. Due to the underlying metric space in our model, there are at most $\mathcal{O}(k^{\delta})$ nodes in S within distance $kd_{\min}$ of node u. Let v be a node at distance at most $2d_{\min}$ from u. For any constant k_0, with probability $p(1-p)^{\mathcal{O}(k_0^{\delta})} = \Omega(p)$, node v is the only node transmitting among all the nodes within distance $k_0 d_{\min}$ from node u. Further, assuming that all nodes at distance greater than $k_0 d_{\min}$ transmit, the interference $I_{S\cup W}(u)$ at u can be bounded from above by

$$
\begin{aligned}
I_{S\cup W}(u) &\leq I_W(u) + \sum_{w\in S\ s.t.\ d(u,w)\geq k_0 d_{\min}} \frac{P}{d(u,w)^{\alpha}} \leq I_W(u) + \sum_{k=k_0}^{\infty} \sum_{w\in S\ s.t.\ 1\leq \frac{d(u,w)}{kd_{\min}} < 1+\frac{1}{k}} \frac{P}{d(u,w)^{\alpha}} \\
&\overset{(1)}{=} I_W(u) + \sum_{k=k_0}^{\infty} \frac{P}{k^{\alpha} d_{\min}^{\alpha}} O\left(\delta k^{\delta-1}\right) = I_W(u) + \frac{P}{d_{\min}^{\alpha}} O\left(\delta \int_{k_0}^{\infty} k^{-(1+\alpha_{\min}-\delta)} dk\right) \\
&= I_W(u) + \frac{P}{d_{\min}^{\alpha}} \mathcal{O}\left(\delta \frac{k_0^{\delta-\alpha_{\min}}}{\alpha_{\min}-\delta}\right) \leq \left(\frac{\varepsilon_{approg}}{\Lambda}\right)^{\Theta(1)} + \kappa(k_0)\frac{P}{d_{\min}^{\alpha}} \\
&\overset{(2)}{\leq} \left(\frac{\varepsilon_{approg}}{\Lambda}\right)^{\Theta(1)} + \kappa(k_0)\frac{P\cdot\Lambda}{R_{1-2\varepsilon}} \\
&\overset{(3)}{\leq} \kappa(k_0/2)\frac{P\cdot\Lambda}{R_{1-2\varepsilon}}.
\end{aligned}
$$

Step (1) stems from bounding $|\{w \in S : kd_{\min} \leq d(u, w) < (k + 1)d_{\min}\}|$, the maximum number of nodes within a ring of diameter $d_{\min}$ at distance $kd_{\min}$. If we define the function κ so as to replace the $\mathcal{O}$-term with $\kappa(k_0) = \kappa(k_0, \alpha_{\min}, \delta) > 0$, which decreases polynomially in k_0. Step (2) stems from bounding $I_W(u)$ using Lemma 10.3 and restating $\kappa(k_0)\frac{P}{d_{\min}}$ by $\kappa(k_0)\frac{P\cdot\Lambda}{R_{1-2\varepsilon}}$. Step (3) is true, as $\Lambda \geq 1$, $R_{1-2\varepsilon}$ is constant, and the exponent hidden in the Θ-notation can be chosen arbitrary large in order to match the function κ.

Due to the choice of k_0 and κ, we get for $SINR(u, v, I)$, where I is the set of all nodes with distance greater than $k_0 d_{\min}$:

$$\frac{\frac{P}{d(u,v)^\alpha}}{N + \kappa(k_0)\frac{P}{d_{\min}^\alpha}} \geq \frac{\frac{P}{(2d_{\min})^\alpha}}{N + \kappa(k_0)\frac{P}{d_{\min}^\alpha}} \geq \frac{\frac{P}{r_s^\alpha}}{\frac{P}{\beta r_w^\alpha} + \kappa(k_0)\frac{2^\alpha P}{r_s^\alpha}} = \frac{\beta}{\frac{1}{(1+\rho)^\alpha} + \kappa(k_0)\beta 2^\alpha} \geq \beta$$

The second inequality follows from $N = \frac{p}{\beta r_w^\alpha}$ and from $d_{\min} \leq r_s/2$. The last inequality holds for sufficiently large k_0. If we choose μ to be the probability that no more than one node in a ball of radius $k_0 d_{\min}$ transmits, then node v can transmit to u with probability μ.

In the above proof, μ depends on the unknown parameter β, so we use β_{max} as the base for computing μ. Note also that since $H_p^\mu[S] \subseteq \tilde{H}_p^\mu[S]$, the lemma induces the same properties on $\tilde{H}_p^\mu[S]$ with high probability. □

Lemma 10.15 (Version of Lemma 4.4 of [13])**.** *Given node $i \in N_{G_{1-2\varepsilon}}(S_1)$ and assume Properties 1 and 2 of Definition 10.8 of a successful epoch at point i are satisfied. Then for any $\phi \in \{1, \ldots, \Phi\}$, the minimum distance between any two nodes in $S_{\phi,i}$ is at least $d_\phi \geq 2^{\phi-1} \cdot d_{\min}$.*

Proof. The proof appears in the full version of this paper [25], as it requires only a minimal modification of a proof provided in [13]. □

Lemma 10.16 (Extended version of Lemma 4.5, [13])**.** *For all $p \in (0, 1/2]$, there is a $\hat{Q}, \gamma = \Theta(1)$, such that for all $Q \geq \hat{Q}$ the following holds. Consider a round r in phase ϕ where each node in S_ϕ transmits a bcast-message with probability p/Q (Line 11). Let $i \in N_{G_{1-2\varepsilon}}(S_1)$ and let $u_\phi \in S_\phi \setminus \{v\}$ be the closest node to v in S_ϕ. Assume Property 1 of Definition 10.8 of a successful epoch at point i are satisfied. Let d_{u_ϕ} be the distance between u_ϕ and its farthest neighbor in $\tilde{\tilde{H}}_p^\mu[S_\phi]$. If $d(u_\phi, v) \leq (1+\varepsilon)R_{1-2\varepsilon}$ and $d_{u_\phi} \geq \gamma Q^{-1/\alpha} \cdot d(u_\phi, v)$, node v receives a bcast-message from u_ϕ in round r with probability $\Theta(1/Q)$.*

Note that the proof presented in [13] reveals that the bcast-message is actually received from node u_ϕ such that we adapted the statement to this fact (instead of stating that v receives a message from some node). We restate the full proof of [13] with our extensions that yield Lemma 10.16. There are two main issues we need to take care of:

1. The proof presented in [13] relies on $\tilde{H}_p^\mu[S_\phi]$ being a γ-close approximation of $H_p^\mu[S_\phi]$. When looking at this proof in more detail, it turns out that this approximation is only required for all nodes located at distance at most $2d_{u_\phi}$ around u_ϕ. We show that this area is covered by a $\mathcal{O}(1)$-neighborhood of u_ϕ in $\tilde{\tilde{H}}_p^\mu[S_\phi]$ such that the statement of [13] on $\tilde{H}_p^\mu[S_\phi]$ can be transferred to our graph $\tilde{\tilde{H}}_p^\mu[S_\phi]$.

2. The proof given in [13] deals with interference from nodes in S_ϕ at distance further than $2d_\phi$ from u_ϕ. We show that additional interference from nodes W that arises due to our modification of their algorithm is negligible compared to interference from nodes in S_ϕ. We conclude that nodes in W do not affect the remainder of the proof of [13].

of Lemma 10.16. The full proof by [13] with the described extensions is deferred to the Appendix, Lemma C.2 □

Lemma 10.17 (Version of Lemma 4.6. of [13])**.** *Assume Property 2 of the Definition 10.8 of a successful epoch is satisfied. With probability* $1-\varepsilon_{approg}/3$*, either* u*'s bcast-message reaches* i *in phase* ϕ*, or* $d(u_{\phi+1}, i) \le R_{1-2\varepsilon}\left(1+\phi\frac{\varepsilon}{\log\Lambda}\right)$.

Proof. We defer the proof to the Appendix (Lemma C.3), as it requires only a minimal modification of a proof provided in [13]. $\square$

10.5.1 Proof of Lemma 10.13

Proof. We extend and adapt parts of the proof presented in Section 4.3 of [13] to our setting. We show that $\Phi = \Theta(\log\Lambda)$ phases are sufficient such that each node $v \in N_{G_{1-2\varepsilon}}(S_1)$ receives a bcast-message from a $G_{1-\varepsilon}$-neighbor. First, we know due to Lemma 10.15 that the minimum distance between nodes in $S_{\phi,i}$ grows exponentially with ϕ. Therefore in some phase $\phi \le \Phi = \Theta(\log\Lambda)$ (assuming $\Lambda \ge R_{1-2\varepsilon}/d_{\min}$, which is satisfied in any non-trivial instance) the minimum distance between nodes in S_ϕ exceeds $R_{1-2\varepsilon}\cdot\left(1+\phi\frac{\varepsilon}{\log\Lambda}\right)$. Second, by applying Lemma 10.17, there must be a phase ϕ in which i receives with probability $1-\varepsilon_{approg}/3$ a bcast-message from a node $u_\phi \in S_\phi$ at distance $d(u_{\phi+1}, i) \le R_{1-2\varepsilon}\left(1+\phi\frac{\varepsilon}{\log\Lambda}\right)$ to i. As $\phi \le \Phi = \Theta(\log\Lambda)$, this can be bounded to be less than $R_{1-2\varepsilon-2\varepsilon^2} < R_{1-2\varepsilon}$. We conclude that the there must be a bcast-message m' sent by a $G_{1-\varepsilon}$-neighbor which arrives at node i during the epoch. This is the bcast-message m' for which node i outputs an *rcv* event (Line 11). $\square$

10.6 Runtime of an Epoch

Lemma 10.18. *The runtime of an epoch is*

$$\mathcal{O}\left(\log^{\alpha+1}(\Lambda)\cdot\log\left(\frac{1}{\varepsilon_{approg}}\right) \quad + \quad \log(\Lambda)\cdot\log\left(\frac{1}{\varepsilon_{approg}}\right)\cdot\log^*\left(\frac{1}{\varepsilon_{approg}}\right)\right).$$

Proof. Due to Lemma 10.10 each execution of Line 8 that constructs a graph $\tilde{\tilde{H}}_p^\mu[S_\phi]$ takes time $\mathcal{O}\left(\Phi+\log(1/\varepsilon_{approg})\right)$. Due to Lemma 10.11 each execution of Line 9 that constructs a set $S_{\phi+1}$ takes time $\mathcal{O}\left((\Phi+\log(1/\varepsilon_{approg}))\log^*(\Lambda/\varepsilon_{approg})\right)$. In Lines 10–13 a bcast-message is sent $\mathcal{O}(Q\log(1/\varepsilon_{approg}))$ times. Due to the choice of Q this is $\mathcal{O}(\log^\alpha(\Lambda)\cdot\log(1/\varepsilon_{approg}))$. All this is executed for each of the Φ phases of an epoch (Lines 6– 15). Thus the total runtime of an epoch is

$$\begin{aligned}
&\mathcal{O}\left(\Phi\left(\left(\Phi \quad + \quad \log\left(\frac{1}{\varepsilon_{approg}}\right)\right)\log^*(\Lambda/\varepsilon_{approg}) \quad + \quad \log^\alpha(\Lambda)\cdot\log\left(\frac{1}{\varepsilon_{approg}}\right)\right)\right)\\
= \quad &\mathcal{O}\Bigg(\log^2(\Lambda)\log^*(\Lambda/\varepsilon_{approg}) \quad + \quad \log(\Lambda)\log^*(\Lambda/\varepsilon_{approg})\log\left(\frac{1}{\varepsilon_{approg}}\right)\\
&\qquad\qquad + \quad \log^{\alpha+1}(\Lambda)\cdot\log\left(\frac{1}{\varepsilon_{approg}}\right)\Bigg)\\
= \quad &\mathcal{O}\left(\log^{\alpha+1}(\Lambda)\cdot\log\left(\frac{1}{\varepsilon_{approg}}\right) \quad + \quad \log(\Lambda)\cdot\log\left(\frac{1}{\varepsilon_{approg}}\right)\cdot\log^*\left(\frac{1}{\varepsilon_{approg}}\right)\right),
\end{aligned}$$

where we use the definition of $\Phi = \Theta(\log\Lambda)$ and $\alpha > 2$. $\square$

10.7 Proof of Theorem 9.1 (Approximate Progress Bound)

Proof. **Probability of approximate progress conditioned on locally unique labels (Remark 10.9).** Consider any node i that has an $G_{1-2\varepsilon}$-neighbor j with an ongoing broadcast event (i.e. $j \in S_1$). Under the assumption of 1) locally unique labels (Remark 10.9), and 2) satisfaction of Properties 1 and 2 of Definition 10.8 of a successful epoch at point i, Lemma 10.13 states that node i receives a bcast-message from a $G_{1-\varepsilon}$-neighbor of i with probability $1 - \varepsilon_{approg}/3$ within one epoch. Lemma 10.12 provides that Properties 1 and 2 of Definition 10.8 of a successful epoch at point i are satisfied with probability $1 - \varepsilon_{approg}/3$. Therefore the total probability that i receives a bcast-message from a $G_{1-\varepsilon}$-neighbor within an epoch is at least $1 - 2\varepsilon_{approg}/3$, which on the one hand implies satisfaction of Property 3 of Definition 10.8 of a successful epoch at point i, and on the other hand implies that approximate progress is made at point i.

Probability of locally unique labels (Remark 10.9). We apply Lemma 10.1 with $H := \tilde{\tilde{H}}_p^\mu[S_\phi]$ and $U := U_{\phi,i}$ using random labels $\in [1, \frac{\text{poly}\,\Lambda}{\varepsilon_{approg}}]$. Lemma 10.1 can be applied, as $|U_{\phi,i}| = \mathcal{O}(\Lambda^2)$. This implies that the modified MIS algorithm computes an independent set that is maximal with respect to $N_{\tilde{\tilde{H}}_p^\mu[S_\phi], c \cdot 4^\Phi \cdot \log^*(\Lambda/\varepsilon_{approg})}(U_{\phi,i})$ with probability $1 - \frac{\varepsilon_{approg}}{3\Phi}$. As $S_{\phi,i} \subseteq N_{\tilde{\tilde{H}}_p^\mu[S_\phi], c \cdot 4^\Phi \cdot \log^*(\Lambda/\varepsilon_{approg})}(U_{\phi,i})$, set $S_{\phi+1}$ is (ϕ, i)-locally maximal in $\tilde{\tilde{H}}_p^\mu[S_\phi]$ with probability $1 - \frac{\varepsilon_{approg}}{3\Phi}$. The probability that we can assume locally unique labels (Remark 10.9) at each of the Φ phases is at least $\left(1 - \frac{\varepsilon_{approg}}{3\Phi}\right)^\Phi \geq 1 - \varepsilon_{approg}/3$.

Final conclusion. When the two arguments the are combined, we conclude that approximate progress is made within one epoch with probability at least $(1 - 2\varepsilon_{approg}/3) \cdot (1 - \varepsilon_{approg}/3) \geq 1 - \varepsilon_{approg}$. Therefore Lemma 10.18 bounds not only the runtime of an epoch, but also f_{approg} by

$$\mathcal{O}\left(\left(\log^\alpha(\Lambda) + \log^*\left(\frac{1}{\varepsilon_{approg}}\right)\right)\log(\Lambda)\log\left(\frac{1}{\varepsilon_{approg}}\right)\right).$$

□

Remark 10.19. *It might be the case that in Algorithm 9.1 a node receives the same bcast-message over and over again for $f_{ack}/2$ time slots (until the sender stops broadcasting), which is an extreme case that still satisfies the definition of progress and approximate progress. We want to stress that this is not a problem, as Algorithm 9.1 is only required to implement fast approximate progress and not acknowledgments. Acknowledgments are obtained in Algorithm B.1.*

11 A Probabilistic AbsMAC Implementation with Fast Acknowledgments and Approximate Progress in the SINR-Model

Theorem 11.1. *Algorithm 11.1 implements the probabilistic absMAC of [37] for $G := G_{1-\varepsilon}$. Approximate progress is measured with respect to $\tilde{G} := G_{1-2\varepsilon}$. The algorithm ensures local* *broadcast in G s.t.*

$$f_{ack} = \mathcal{O}\left(\Delta_{G_{1-\varepsilon}} \cdot \log\left(\frac{\Lambda}{\varepsilon_{ack}}\right) \quad + \quad \log(\Lambda)\log\left(\frac{\Lambda}{\varepsilon_{ack}}\right)\right)$$

with probability at least $1-\varepsilon_{ack}$ with respect to acknowledgments in $G := G_{1-\varepsilon}$, and

$$f_{approg} = \mathcal{O}\left(\left(\log^{\alpha}(\Lambda) + \log^{*}\left(\frac{1}{\varepsilon_{approg}}\right)\right)\log(\Lambda)\log\left(\frac{1}{\varepsilon_{approg}}\right)\right).$$

with probability at least $1-\varepsilon_{approg}$ with respect to approximate progress in $\tilde{G} := G_{1-2\varepsilon}$.

Remark 11.2. *The bound on f_{approg} is significantly better than the best possible bound on f_{prog} due to the lower bound in Theorem 6.1. E.g. for graphs where $\Delta_{G_{1-\varepsilon}} = \Omega(\Lambda^{\gamma})$, for $\gamma > 0$, the bound on f_{approg} of Theorem 11.1 is polylogarithmic in the degree $\Delta_{G_{1-\varepsilon}}$, while the lower bound on f_{prog} in Section 6 is linear in the degree $\Delta_{G_{1-\varepsilon}}$.*

To achieve the bounds stated in Theorem 11.1, we use two algorithms that run in parallel.

- Algorithm of Theorem 5.1 is executed in every even time step with respect to $G_{1-\varepsilon}$ and ensures an almost optimal bound on f_{ack}.
- Algorithm 9.1 is executed in every odd time step and ensures fast approximate progress with respect to $\tilde{G}$.

Combining these two algorithms provides good bounds on both, f_{ack} and f_{approg}. Such a combination is necessary, as the Algorithm of Theorem 5.1 might not yield a good bound on approximate progress and Algorithm 9.1 might not lead to an acknowledgment at all. Therefore they complement each other.

11.1 Details of the Algorithm

Once a $bcast(m)_i$ event occurs at node i, node i starts to execute Algorithm 11.1 for f_{ack} time steps. After this node i performs $ack(m)_i$. If node i has an ongoing broadcast and receives an $abort(m)_i$ input from the environment before it performs $ack(m)_i$, then it (i) continues to participate until the current epoch of Algorithm 9.1 is finished, (ii) after this epoch performs no further transmission on behalf of bcast-message m, and (iii) does not perform $ack(m)_i$. Whenever a node i receives a message m' for the first time in an epoch, it delivers that message to its environment with a $rcv(m')_i$ output event.

Algorithm 11.1 Implementation of local broadcast as executed by a node i.

```
1: m := 0; // m stores bcast-message input of an ongoing bcast-event at i
2: whenever a bcast-message is received or a bcast(m')_i event occurs, wake up if not awake;
3: whenever a bcast(m')_i event occurs, set m := m' and reset m := 0 after f_ack rounds;
4: Execute in parallel in even/odd time steps:
       The algorithm of Theorem 5.1, and
       Algorithm 9.1;
```

of Theorem 11.1. Details on the Algorithm and proof corresponding to the bound on f_{ack} in Theorem 11.1 are stated in Section 5. Details on the Algorithm and proof corresponding to the bound on f_{approg} in Theorem 11.1 are stated in Sections 9 and 10. We also point the reader to Remark 4.6 used in these proofs. $\square$

12 Application: Improved Network-Wide Broadcast

We combine this absMAC implementation with algorithms from [37] for global broadcast in this absMAC. We recall the relevant Theorems of [37]. In Theorem 12.6 we argue that we can replace f_{prog} and ε_{prog} in the relevant Theorems of [37] by f_{approg} and ε_{approg} under certain conditions and state the effect that this replacement has on other parameters of the runtime. We use

$$c_2 := \frac{2}{1-\varepsilon_{prog}} \qquad \textit{and} \qquad c_3 := \frac{3}{1-\varepsilon_{prog}}.$$

For the convenience of the reader we restate Theorem 7.7 and 8.20 of [37] with respect to broadcast in graph G and diameter D_G, and recall notation used in these Theorems.

Theorem 12.1 (Version of Theorem 7.7 of [37])**.** *Let G be a graph in which local broadcast is implemented that can be used via the probabilistic absMAC. Let γ' be a real number, $0 < \gamma' \leq 1$. The BSMB protocol [of [37]] guarantees that, with probability at least*

$$1 - \gamma' - n \cdot \varepsilon_{ack},$$

rcv events and hence, deliver events, occur at all nodes $\neq i_0$ by time

$$(c_3 D_G + c_2 \ln(n/\gamma')) f_{prog}$$

Definition 12.2 (Nice broadcast events and nice executions, Definition 4.1 of [37])**.** *Suppose a $bcast(m)_i$ event π occurs at time t_0 in execution α. Then we say that π is nice if $ack(m)_i$ occurs by time $t_0 + f_{ack}$ and is preceded by a $rcv(m)_j$ for every neighbor j of i. We say that execution α is nice if all bcast events in α are nice. Let N be the set of all nice executions.*

Definition 12.3 (Clear events, Definition 8.1 of [37])**.** *Let α be an execution in N (the set of nice executions), and let $m \in M$ be a message such that an $arrive(m)$ event occurs in α. We define the event $clear(m)$ to be the final $ack(m)$ event in α.*

Definition 12.4 (The Set $K(m)$, Definition 8.2 of [37])**.** *Let α be an execution in N and let $m \in M$ be a message such that $arrive(m)$ occurs in α. We define $K(m)$ to be the set of messages $m' \in M$ such that an $arrive(m')$ event precedes the $clear(m)$ event and the $clear(m')$ event follows the $arrive(m)$ event. That is, $K(m)$ is the set of messages whose processing overlaps the interval between the $arrive(m)$ and $clear(m)$ events.*

Theorem 12.5 (Version of Theorem 8.20 of [37])**.** *Let G be a graph in which local broadcast is implemented that can be used via the probabilistic absMAC. Let $m \in M$ and let γ' be a real number, $0 < \gamma' < 1$. The BMMB protocol [of [37]] guarantees that, with probability at least*

$$1 - \gamma' - nk\varepsilon_{ack},$$

the following property holds for the generated execution α:
Suppose an $arrive(m)_i$ event π occurs in α, and let t_0 be the time of occurrence of π. Let k' be a positive integer such that $|K(m)| \leq k'$. Then $get(m)$ events, and hence, deliver events occur at all nodes in α by time

$$t_0 + \left((c_3 + c_2)D_G + ((c_3 + 2c_2)\left\lceil \ln\left(\frac{2n^3k}{\gamma'}\right)\right\rceil + c_3 + c_2)k'\right) f_{prog} + (k'-1)f_{ack}$$

Theorem 12.6. *Let G be a graph in which local broadcast is available via the probabilistic absMAC of [37]. Let $\tilde{G}$ be the graph in which approximate progress is measured and let the vertex sets of the connected components of $\tilde{G}$ and G be the same. Then one can replace $f_{prog}, \varepsilon_{prog}$ and D_G in Theorems 12.1 and 12.5 concerning their global SMB and MMB algorithms by $f_{approg}, \varepsilon_{approg}$ and $D_{\tilde{G}}$.*

Proof. We start by recalling the BMMB and BSMB protocols of [37] for global MMB and SMB, for which we present our argument.

Basic Multi-Message Broadcast (BMMB) Protocol: Every process i maintains a FIFO queue named *bcastq* and a set named *rcvd*. Both are initially empty. If process i is not currently sending a message on the MAC layer and its *bcastq* is not empty, it sends the message at the head of the queue on the MAC layer (disambiguated with identifier i and sequence number) using a *bcast* output. If i receives a message from the environment via an $arrive(m)_i$ input, it immediately delivers the message m to the environment using a $deliver(m)_i$ output, and adds m to the back of *bcastq* and to the *rcvd* set. If i receives a message m from the MAC layer via a $rcv(m)_i$ input, it first checks *rcvd*. If $m \in rcvd$ it discards it. Else, i immediately performs a $deliver(m)_i$ output and adds m to *bcastq* and *rcvd*.

Basic Single-Message Broadcast (BSMB) Protocol: This is just BMMB specialized to one message, and modified so that the message starts in the state of a designated initial node i_0.

In the above algorithms, once a node i receives a message, node i broadcasts the message if it did not broadcast it before. The result of global broadcast is independent of whether a message was received due to transmission from a $\tilde{G}$-neighbor or a G-neighbor as long as the components of $\tilde{G}$ and G are the same. Only the runtime changes.

In time f_{prog} it is guaranteed that with probability $1 - \varepsilon_{prog}$ a message is received by a node v when a G-neighbor of v is sending. Therefore the runtime presented in [37] depends on $D_{\tilde{G}}$. Compared to this it is guaranteed with probability $1 - \varepsilon_{approg}$ that in time f_{approg} a message arrives when a $\tilde{G}$-neighbor is sending. A message that causes approximate progress in G with respect to $\tilde{G}$ essentially causes progress in $\tilde{G}$ if we restrict local broadcast to $\tilde{G}$. Here, if required by the specification of the abstract MAC layer (or algorithms using it) we output *rcv*-events for messages that arrive from G-neighbors, but not from other nodes outside of G. Therefore D_G needs to be replaced by $D_{\tilde{G}}$.

Now note that a node i that receives a message m from the MAC layer via a $rcv(m)_i$ discards m if $m \in rcvd$. Therefore messages from G are only placed into *bcastq* once and cannot cause delays more than once. Based on this we can now replace f_{prog} and ε_{prog} in Theorems 12.1 and 12.5 by f_{approg} and ε_{approg} if we also take into account the change of the diameter of the graph in which we consider broadcast. Therefore the diameter D_G is replaced by $D_{\tilde{G}}$.

Although one might now only need f_{ack} with respect to broadcast in $\tilde{G}$, we still need to use the bound of f_{ack} for G, as broadcast is implemented in G. We conclude the statement, as $G := G_{1-\varepsilon}$ and $\tilde{G} := G_{1-2\varepsilon}$. $\square$

By combining Theorems 12.1 and 12.5 with our results, we obtain:

Theorem 12.7. *Consider the SINR model using the model assumptions stated in Section 4.6. We present an algorithm that performs global SMB in graph $G_{1-\varepsilon}$ with probability at least $1-\varepsilon_{SMB}$ in time*

$$\mathcal{O}\left(\left(D_{G_{1-2\varepsilon}} + \log\left(\frac{n}{\varepsilon_{SMB}}\right)\right) \cdot \log^{\alpha+1}(\Lambda)\right).$$

$$\mathcal{O}\left(D_{G_{1-2\varepsilon}} \log^{\alpha+1}(\Lambda) \quad + \quad k'\left(\Delta_{G_{1-\varepsilon}} + \mathit{polylog}\left(\frac{nk\Lambda}{\varepsilon_{MMB}}\right)\right) \log\left(\frac{nk}{\varepsilon_{MMB}}\right)\right).$$

Proof. Theorem 11.1 states that $f_{ack} = \mathcal{O}\left(\Delta_{G_{1-\varepsilon}} \cdot \log\left(\frac{\Lambda}{\varepsilon_{ack}}\right) \quad + \quad \log(\Lambda)\log\left(\frac{\Lambda}{\varepsilon_{ack}}\right)\right)$ and $f_{approg} = \mathcal{O}\left(D_{G_{1-2\varepsilon}} \log^{\alpha+1}(\Lambda) \quad + \quad k(\Delta_{G_{1-\varepsilon}} + \mathrm{polylog}\,(nk\Lambda)) \log\,(nk)\right)$.

Global SMB: Theorem 12.1 combined with Theorem 12.6 guarantees that for $0 < \gamma' \leq 1$ with probability $1 - \gamma' - n \cdot \varepsilon_{ack}$, global SMB can be performed in time $(c_3 D_{G_{1-2\varepsilon}} + c_2 \ln(n/\gamma'))f_{approg}$. We choose $\gamma' = \varepsilon_{SMB}/2$ and $\varepsilon_{ack} := \varepsilon_{SMB}/(2n)$. Therefore we obtain that global SMB is performed with probability $1 - \gamma' - n \cdot \varepsilon_{ack} = 1 - \varepsilon_{SMB}$. Choosing $\varepsilon_{approg} := 1/8$ yields the following total runtime:

$$(c_3 D_{G_{1-2\varepsilon}} + c_2 \ln(n/\gamma'))f_{approg} = \mathcal{O}\left((D_{G_{1-2\varepsilon}} + \log(n/\varepsilon_{SMB})) \cdot \log^{\alpha+1}(\Lambda)\right).$$

Global MMB: Theorem 12.5 guarantees that for $0 < \gamma' \leq 1$ with probability $1-\gamma'-nk\varepsilon_{ack}$, global MMB is completed at time

$$t_0 + \left((c_3 + c_2)D_{G_{1-2\varepsilon}} + ((c_3 + 2c_2)\left\lceil \ln\left(\frac{2n^3k}{\gamma'}\right)\right\rceil + c_3 + c_2)k'\right) f_{prog} + (k'-1)f_{ack}.$$

We choose $\gamma' = \varepsilon_{MMB}/(2k)$ and $\varepsilon_{ack} := \varepsilon_{MMB}/(2kn)$. Therefore we obtain that global MMB is performed with probability $1 - \gamma' - nk \cdot \varepsilon_{ack} = 1 - \varepsilon_{MMB}$. This yields the following total runtime:

$$\begin{aligned}
&\left((c_3 + c_2)D_{G_{1-2\varepsilon}} + ((c_3 + 2c_2)\left\lceil \ln\left(\frac{2n^3k}{\gamma'}\right)\right\rceil + c_3 + c_2)k'\right) f_{prog} + (k'-1)f_{ack} \\
&= \mathcal{O}\left(D_{G_{1-2\varepsilon}} f_{prog} + k'(f_{ack} + \log\,(nk/\varepsilon_{MMB})\, f_{prog})\right) \\
&= \mathcal{O}\left(D_{G_{1-2\varepsilon}} f_{approg} + k'(f_{ack} + \log\,(nk/\varepsilon_{MMB})\, f_{approg})\right) \\
&= \mathcal{O}\left(D_{G_{1-2\varepsilon}} \log^{\alpha+1}(\Lambda) \quad + \quad k'\left(\Delta_{G_{1-\varepsilon}} + \mathrm{polylog}\left(\frac{nk\Lambda}{\varepsilon_{MMB}}\right)\right) \log\left(\frac{nk}{\varepsilon_{MMB}}\right)\right)
\end{aligned}$$

In our setting we can simply replace k' by k, as we consider the one-shot version of k-message broadcast. This results in the claimed runtime. Furthermore note that we do not need the model assumption (see Section 4.6) that nodes know their $G_{1-\varepsilon}$-neighbors in case $G_{1-\varepsilon}$ is connected (also see the discussion in Remark 4.6). When looking at the proof of Theorem 12.6 and the BMMB protocol stated therein, we conclude that even if messages are received from nodes in transmission range that are not $G_{1-\varepsilon}$-neighbors, messages are added to *bcastq* only once and cannot cause delays several times. □

Acknowledgments: We thank Sebastian Daum, Mohsen Ghaffari, Fabian Kuhn and Calvin Newport for answering questions concerning their earlier work and helpful discussions. In particular we thank Erez Kantor for many helpful discussions—especially in the early stages of this work.

References

[1] K. Alekeish and P. D. Ezhilchelvan. Consensus in sparse, mobile ad hoc networks. *IEEE Trans. Parallel Distrib. Syst.*, pages 467–474, 2012.

[2] A. Alon, A. Bar-Noy, N. Linial, and D. Peleg. A lower bound for radio broadcast. *J. Compt. Syst. Science*, 43:290–298, 1991.

[3] M. Andrews and M. Dinitz. Maximizing capacity in arbitrary wireless networks in the SINR model: Complexity and game theory. In *Proc. of the 28th International Conference on Computer Communications, INFOCOM, Rio de Janeiro*, pages 1332–1340. IEEE, 2009.

[4] R. Bar-Yehuda, O. Goldreich, and A. Itai. On the time-complexity of broadcast in multi-hop radio networks: An exponential gap between determinism and randomization. *Journal of Computer and System Sciences*, 45(1):104–126, 1992.

[5] R. Bar-Yehuda, A. Israeli, and A. Itai. Multiple communication in multihop radio networks. *SIAM J. Comput.*, 22(4):875–887, 1993.

[6] M. H. L. Bodlaender, M. M. Halldórsson, and P. Mitra. Connectivity and aggregation in multihop wireless networks. In *ACM Symposium on Principles of Distributed Computing, PODC '13, Montreal, QC, Canada, July 22-24, 2013*, pages 355–364, 2013.

[7] I. Chlamtac and S. Kutten. On broadcasting in radio networks–problem analysis and protocol design. *IEEE Transactions on Communications*, 33(12):1240–1246, 1985.

[8] I. Chlamtac and O. Weinstein. The wave expansion approach to broadcasting in multihop radio networks. *IEEE Transactions on Communications*, 39:426–433, 1991.

[9] G. Chockler, M. Demirbas, S. Gilbert, C. C. Newport, and T. Nolte. Consensus and collision detectors in wireless ad hoc networks. In M. K. Aguilera and J. Aspnes, editors, *Proceedings of the Twenty-Fourth Annual ACM Symposium on Principles of Distributed Computing, PODC 2005, Las Vegas, NV, USA, July 17-20, 2005*, pages 197–206, 2005.

[10] A. Cornejo, N. Lynch, S. Viqar, and J. L. Welch. Neighbor discovery in mobile ad hoc networks using an abstract mac layer. In *Communication, Control, and Computing, 2009. Allerton 2009. 47th Annual Allerton Conference on*, pages 1460–1467. IEEE, 2009.

[11] A. Cornejo, S. Viqar, and J. L. Welch. Reliable neighbor discovery for mobile ad hoc networks. *Ad Hoc Networks*, 12:259–277, 2014.

[12] A. Czumaj and W. Rytter. Broadcasting algorithms in radio networks with unknown topology. *J. Algorithms*, 60(2):115–143, 2006.

[13] S. Daum, S. Gilbert, F. Kuhn, and C. Newport. Broadcast in the ad hoc SINR model. Technical Report 274, University of Freiburg, Dept. of Computer Science, 2013.

[14] S. Daum, S. Gilbert, F. Kuhn, and C. C. Newport. Broadcast in the ad hoc SINR model. In *Distributed Computing - 27th International Symposium, DISC 2013, Jerusalem, Israel, October 14-18, 2013. Proceedings*, pages 358–372, 2013.

[15] M. Dinitz. Distributed algorithms for approximating wireless network capacity. In *INFOCOM 2010. 29th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 15-19 March 2010, San Diego, CA, USA*, pages 1397–1405, 2010.

[16] M. Elkin and G. Kortsarz. Improved schedule for radio broadcast. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2005, Vancouver, British Columbia, Canada, January 23-25, 2005*, pages 222–231, 2005.

[17] G. Even, Y. Matsri, and M. Medina. Multi-hop routing and scheduling in wireless networks in the SINR model. In *Algorithms for Sensor Systems - 7th International Symposium on Algorithms for Sensor Systems, Wireless Ad Hoc Networks and Autonomous Mobile Entities, ALGOSENSORS 2011, Saarbrücken, Germany, September 8-9, 2011, Revised Selected Papers*, pages 202–214, 2011.

[18] I. Gaber and Y. Mansour. Centralized broadcast in multihop radio networks. *Journal of Algorithms*, 46:(1),1–20, 2003.

[19] L. Gasieniec, D. Peleg, and Q. Xin. Faster communication in known topology radio networks. *Distributed Computing*, 19(4):289–300, 2007.

[20] M. Ghaffari, B. Haeupler, and M. Khabbazian. A bound on the throughput of radio networks. *CoRR*, abs/1302.0264, 2013.

[21] M. Ghaffari, B. Haeupler, and M. Khabbazian. Randomized broadcast in radio networks with collision detection. In *ACM Symposium on Principles of Distributed Computing, PODC '13, Montreal, QC, Canada, July 22-24, 2013*, pages 325–334, 2013.

[22] M. Ghaffari, B. Haeupler, N. A. Lynch, and C. C. Newport. Bounds on contention management in radio networks. In M. K. Aguilera, editor, *Distributed Computing - 26th International Symposium, DISC 2012, Salvador, Brazil, October 16-18, 2012. Proceedings*, volume 7611 of *Lecture Notes in Computer Science*, pages 223–237. Springer, 2012.

[23] M. Ghaffari, E. Kantor, N. Lynch, and C. Newport. Multi-message broadcast with abstract MAC layers and unreliable links. In *Proceedings of the 33rd Annual ACM Symposium on Principles of Distributed Computing, PODC 2014, Paris, France, July 15-18, 2014*, pages 56–65, extended version available at http://arxiv.org/abs/1405.1671, 2014.

[24] O. Goussevskaia, T. Moscibroda, and R. Wattenhofer. Local broadcasting in the physical interference model. In *DIALM-POMC*, pages 35–44, 2008.

[25] M. M. Halldorsson, S. Holzer, and N. Lynch. A local broadcast layer for the sinr network model. *arXiv preprint arXiv:1505.tba*, 2015.

[26] M. M. Halldórsson, S. Holzer, P. Mitra, and R. Wattenhofer. The power of non-uniform wireless power. In S. Khanna, editor, *Proceedings of the 24th annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1595–1606, 2013.

[27] M. M. Halldórsson and P. Mitra. Nearly optimal bounds for distributed wireless scheduling in the SINR model. In L. Aceto, M. Henzinger, and J. Sgall, editors, *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zürich, Switzerland, July 4-8, 2011, Proceedings, Part II*, volume 6756 of *Lecture Notes in Computer Science*, pages 625–636. Springer, Berlin & Heidelberg, Germany, 2011.

[28] M. M. Halldórsson and P. Mitra. Distributed connectivity of wireless networks. In D. Kowalski and A. Panconesi, editors, *Proceedings of the 31st annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing, PODC 2012, Funchal, Madeira, Portugal, July 16-18, 2012*, pages 205–214, 2012.

[29] M. M. Halldórsson and P. Mitra. Towards tight bounds for local broadcasting. In *FOMC'12, The Eighth ACM International Workshop on Foundations of Mobile Computing (part of PODC 2012), Funchal, Portugal, July 19, 2012, Proceedings*, page 2, 2012.

[30] N. Hobbs, Y. Wang, Q.-S. Hua, D. Yu, and F. C. M. Lau. Deterministic distributed data aggregation under the SINR model. In M. Agrawal, S. B. Cooper, and A. Li, editors, *TAMC*, volume 7287 of *Lecture Notes in Computer Science*, pages 385–399. Springer, Heidelberg, 2012.

[31] T. Jurdzinski, D. R. Kowalski, M. Rozanski, and G. Stachowiak. Distributed randomized broadcasting in wireless networks under the SINR model. In *Distributed Computing*, pages 373–387. 2013.

[32] T. Jurdzinski, D. R. Kowalski, M. Rozanski, and G. Stachowiak. On the impact of geometry on ad hoc communication in wireless networks. In M. M. Halldórsson and S. Dolev, editors, *Proceedings of the Thirty-Third Annual ACM Symposium on Principles of Distributed Computing, PODC '14, Paris, France, July 15-18, 2014*, pages 357–366, 2014.

[33] T. Jurdzinski, D. R. Kowalski, and G. Stachowiak. Distributed deterministic broadcasting in wireless networks of weak devices. In *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part II*, pages 632–644, 2013.

[34] T. Kesselheim. A constant-factor approximation for wireless capacity maximization with power control in the SINR model. In D. Randall, editor, *Proceedings of the Twenty-Second*

Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011, pages 1549–1559, 2011.

[35] T. Kesselheim and B. Vöcking. Distributed contention resolution in wireless networks. In N. A. Lynch and A. A. Shvartsman, editors, *Distributed Computing (proceedings of DISC)*, volume 6343 of *Lecture Notes in Computer Science*, pages 163–178. Springer, Heidelberg, 2010.

[36] M. Khabbazian and D. R. Kowalski. Time-efficient randomized multiple-message broadcast in radio networks. In C. Gavoille and P. Fraigniaud, editors, *Proceedings of the 30th Annual ACM Symposium on Principles of Distributed Computing, PODC 2011, San Jose, CA, USA, June 6-8, 2011*, pages 373–380, 2011.

[37] M. Khabbazian, D. R. Kowalski, F. Kuhn, and N. A. Lynch. Decomposing broadcast algorithms using abstract MAC layers. *Ad Hoc Networks*, 12:219–242, 2014.

[38] M. Khabbazian, F. Kuhn, N. A. Lynch, M. Médard, and A. ParandehGheibi. MAC design for analog network coding. In A. Chaintreau and D. R. Kowalski, editors, *FOMC'11, The Seventh ACM SIGACT/SIGMOBILE International Workshop on Foundations of Mobile Computing (part of FCRC 2011), San Jose, CA, USA, June 9, 2011, Proceedings*, pages 42–51, 2011.

[39] D. Kowalski and A. Pelc. Optimal deterministic broadcasting in known topology radio networks. *Distributed Computing*, 19:185–195, 2007.

[40] D. R. Kowalski and A. Pelc. Broadcasting in undirected ad hoc radio networks. *Distributed Computing*, 18(1):43–57, 2005.

[41] F. Kuhn, N. A. Lynch, and C. C. Newport. The abstract MAC layer. *Distributed Computing*, 24(3-4):187–206, 2011.

[42] E. Kushilevitz and Y. Mansour. An $\Omega(D \log(N/D))$ lower bound for broadcast in radio networks. *SIAM J. on Computing*, 27:702–712, 1998.

[43] T. Moscibroda and R. Wattenhofer. The Complexity of Connectivity in Wireless Networks. In *Proceedings of the 25th annual IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2006, Barcelona, Catalunya, Spain, 23-29 April 2006*, pages 1–13, 2006.

[44] C. C. Newport. Consensus with an abstract MAC layer. In M. M. Halldórsson and S. Dolev, editors, *ACM Symposium on Principles of Distributed Computing, PODC '14, Paris, France, July 15-18, 2014*, pages 66–75, 2014.

[45] D. Peleg. Time-efficient broadcasting in radio networks: A review. In T. Janowski and H. Mohanty, editors, *Distributed Computing and Internet Technology, 4th International Conference, ICDCIT 2007, Bangalore, India, December 17-20, Proceedings*, pages 1–18, 2007.

[46] C. Scheideler, A. W. Richa, and P. Santi. An O(log n) dominating set protocol for wireless ad-hoc networks under the physical interference model. In *MobiHoc*, pages 91–100, 2008.

[47] J. Schneider and R. Wattenhofer. A log-star distributed maximal independent set algorithm for growth-bounded graphs. In R. A. Bazzi and B. Patt-Shamir, editors, *Proceedings of the Twenty-Seventh Annual ACM Symposium on Principles of Distributed Computing, PODC 2008, Toronto, Canada, August 18-21, 2008*, pages 35–44, 2008.

[48] D. Yu, Q. Hua, Y. Wang, and F. C. M. Lau. An o(log n) distributed approximation algorithm for local broadcasting in unstructured wireless networks. In *IEEE 8th International Conference on Distributed Computing in Sensor Systems, DCOSS 2012, Hangzhou, China, 16-18 May, 2012*, pages 132–139, 2012.

[49] D. Yu, Q.-S. Hua, Y. Wang, H. Tan, and F. C. M. Lau. Distributed multiple-message broadcast in wireless ad-hoc networks under the SINR model. In G. Even and M. M. Halldórsson, editors, *Structural Information and Communication Complexity - 19th International Colloquium, SIROCCO 2012, Reykjavik, Iceland, June 30-July 2, 2012, Revised Selected Papers*, volume 7355 of *Lecture Notes in Computer Science*, pages 111–122. Springer, Heidelberg, 2012.

[50] D. Yu, Q.-S. Hua, Y. Wang, J. Yu, and F. C. M. Lau. Efficient distributed multiple-message broadcasting in unstructured wireless networks. In *Proceedings of the IEEE INFOCOM 2013, Turin, Italy, April 14-19, 2013*, pages 2427–2435, 2013.

Appendix

A Basic Lemma on Growth Bounded Graphs

Lemma A.1. *Let G be growth bounded with polynomial bounding function $f(r)$. Then it is $|N_{G,r}(v)| \leq \Delta f(r)$.*

Proof. This statement is well known in the unit-disc graph community. As we did not find a reference to this version of the statement we include a proof for completeness. The number of nodes in $|N_{G,r}(v)|$ that are in an independent set of G is bounded by $f(r)$. Consider the subgraph H of G that consist of nodes $N_{G,r}(v)$ and edges of G between them. Any independent set in H can be extended to an independent set in G and thus is of size at most $f(r)$. On the other hand an independent set on H dominates all nodes in H such that the size of H is at most $\Delta f(r)$, as each node has degree at most Δ. □

B Proof of [29] Adapted to our Theorem 5.1

We only restate Algorithm and Analysis from [29] adapted to our needs for completeness and convenience of the reader with the goal of making it simpler to verify our claim in Theorem 5.1. The proof is minimal modified and variables are replaced by more general parameters to demonstrate correctness. In particular we restate Theorem 3 of [29] with respect to an upper bound $\tilde{N}_x$ of the local contention N_x. Here N_x is defined to be the number of $G_{(1-\varepsilon)}$-neighbors of $x \in V$ that have ongoing broadcasts at the time of execution. Compared to this Theorem 3 of [29] assumes n as an upper bound such that the runtime depends on n. However, we are interested in local parameters. Furthermore we only wish to claim successful local broadcast within the stated time with probability $1 - \varepsilon_{ack}$, while [29] claims w.h.p., which affects the runtime as well. For simplicity we use their Notation of regions T_x and B_x, see Definition B.1, which at the same time serve as sets of nodes with ongoing broadcast located in these regions. Our notion of $N_{G_(1-\varepsilon)}(x)$ describes a similar set of all nodes in the ball of radius $R_(1-\varepsilon)$, but also those nodes with no ongoing broadcasts. Furthermore this set cannot be treated as an area and B_x is more useful for this.

Definition B.1. *The* transmission region *T_x is the ball of radius R_1 around a node x which x can reach without any other node transmitting. The* broadcasting region *B_x is a ball of radius $R_{(1-\varepsilon)}$ around any node x, containing all nodes to which x would like to transmit.*

Remark B.2. *The analysis below is transferred from [29] and requires ε large enough such that $(1-\varepsilon) =: \phi \leq \frac{1}{6}$. However, this is only for simplicity of the presentation and can be adapted to arbitrary small constant ε.*

Theorem B.3 (Version of Theorem 3 of [29])**.** *Let $\tilde{N}_x$ be an upper bound on the local contention N_x and let $\varepsilon_{ack} > 0$. When executing Algorithm B.1 each node x successfully performs a local broadcast within*

$$O(N_x \log(\tilde{N}_x/\varepsilon_{ack}) + \log(\tilde{N}_x)\log(\tilde{N}_x/\varepsilon_{ack}))$$

rounds with probability at least $1 - \varepsilon_{ack}$.

Algorithm B.1 LocalBroadcast (For any node y)

$tp_y \leftarrow 0$
$p_y \leftarrow \frac{1}{4\tilde{N}_x}$
loop
  $p_y \leftarrow \max\{\frac{1}{128\tilde{N}_x}, \frac{p_y}{32}\}$
  $rc_y \leftarrow 0$
  loop
    $p_y \leftarrow \min\{\frac{1}{16}, 2p_y\}$
    for $j \leftarrow 1, 2, \ldots, \delta \log(\tilde{N}_x/\varepsilon_{ack})$ **do**
      $s \leftarrow 1$ with probability p_y
      if s = 1 **then**
        transmit
      end if
      $tp_y \leftarrow tp_y + p_y$
      if $tp_y > \gamma' \log(\tilde{N}_x/\varepsilon_{ack})$ **then**
        halt;
      end if
      if message received **then**
        $rc_y \leftarrow rc_y + 1$
        if $rc_y > 8\log(2\tilde{N}_x/\varepsilon_{ack})$ **then**
          goto line 4
        end if
      end if
    end for
  end loop
end loop

The symbols γ', λ used in Algorithm B.1 are appropriate constants.

The intuition behind the algorithm is as follows. The “right” probability for x to transmit at is about $\frac{1}{\tilde{N}_x}$ (too high, and collisions are inevitable; too low, nothing happens). The algorithm starts from a low probability, continuously increasing it, but once it starts receiving messages from others, it uses that as an indication that the “right” transmission probability has been reached.

To prove Thm. B.3, we will first need the following definition.

Definition B.4. *For any node x, the event* LowPower *occurs at a time slot if the received power at x from other nodes, $P_x \leq \frac{1}{(4(\beta+4)R_{(1-\varepsilon)})^{\alpha}}$.*

The following technical Lemma follows from geometric arguments.

Lemma B.5. *f x transmits and* LowPower *occurs at x, all nodes in $2B_x$ receive the message from x (thus a successful local broadcast occurs for x).*

Proof. [of Lemma B.5] Consider any $y \in 2B_x$. By definition of $2B_x$, $d(x,y) \leq 2R_{(1-\varepsilon)}$. Now consider any other transmitting node z. We will show that,

Claim B.6. $d(z,x) \leq 3(\beta+2)d(z,y)$

Proof. By the signal propagation model, $\frac{1}{d(z,x)^\alpha}$ is the power received at x from z. Since LowPower occurred,

$$\frac{1}{d(z,x)^\alpha} \leq \frac{1}{((4\beta+4)R_{(1-\varepsilon)})^\alpha}$$
$$\Rightarrow \; d(z,x) \geq 4(\beta+4)R_{(1-\varepsilon)}$$

By the triangle inequality, $d(z,y) \geq d(z,x) - d(x,y) > 4(\beta+4)R_{(1-\varepsilon)} - 2R_{(1-\varepsilon)} \geq 3(\beta+4)R_{(1-\varepsilon)}$, proving the claim. □

This implies, by basic computation and summing over all transmitting z, that

$$P_y \leq \left(\frac{4}{3}\right)^\alpha P_x \tag{3}$$

Now, the SINR at node y (in relation to the message sent by x) is

$$\frac{\frac{1}{2^\alpha R^\alpha_{(1-\varepsilon)}}}{P_y + N} \stackrel{1}{\geq} \frac{\frac{1}{2^\alpha R^\alpha_{(1-\varepsilon)}}}{\left(\frac{4}{3}\right)^\alpha P_x + N} \stackrel{2}{\geq} \frac{\frac{1}{2^\alpha R^\alpha_{(1-\varepsilon)}}}{\left(\frac{4}{3}\right)^\alpha \frac{1}{((4(\beta+4))R_{(1-\varepsilon)})^\alpha} + \frac{(1-\varepsilon)^\alpha}{R^\alpha_{(1-\varepsilon)}\beta}} \stackrel{3}{\geq} \beta$$

Explanation of numbered (in)equalities:

1. By Eqn. 3.
2. Plugging in the bound of P_x (since LowPower occurs at x) and noting that $N = \frac{1}{\beta R_1^\alpha} = \frac{(1-\varepsilon)^\alpha}{\beta R^\alpha_{(1-\varepsilon)}}$, from the definitions of R_1 and $R_{(1-\varepsilon)}$.
3. Follows from simple computation once $(1-\varepsilon)$ is set to a small enough constant ($(1-\varepsilon) = \frac{1}{6}$ suffices).

Thus the SINR condition is fulfilled, and y receives the message from x. □

We will also need the following definition:

Definition B.7. *A* FallBack *event is said to occur for node y if line 20 is executed for y.*

We will refer to the transmission probability p_y for a node y at given time slots. This will always refer to the value of p_y in line 9. We first provide a few basic lemmas needed for the proof of Lemma B.10, that bounds the transmission probability in any broadcast region at a given time.

Lemma B.8. *Consider any slot t and any node z. Assume that in that slot, for all broadcast regions B_x, $\sum_{y \in B_x} \leq \frac{1}{2}$. Then,* LowPower *occurs for z with probability at least $\frac{1}{2}\left(\frac{1}{4}\right)^{\frac{1}{2}O\left(\frac{1}{(1-\varepsilon)^2}\right)}$.*

Proof. Let $B = B_x \setminus \{x\}$. We first prove that there is a substantial probability that no node in B transmits. Assuming this probability is $\mathbb{P}_{N_x}$

$$\mathbb{P}_{N_x} \geq \prod_{w \in B} (1 - p_w) \geq \prod_{w \in B_x} (1 - p_w) \geq \left(\frac{1}{4}\right)^{\sum_w p_w} \geq \left(\frac{1}{4}\right)^{\frac{1}{2}}$$

The third inequality is from Fact 3.1 [24], and the last from the bound $\sum_w p_w \leq \frac{1}{2}$.

Let $\mathbb{P}_T$ be the probability that no other node transmits in T_x. Since $R_{(1-\varepsilon)} = (1-\varepsilon)R_1$, T_x can be covered by $O(\frac{1}{(1-\varepsilon)^2})$ broadcast regions (this can be shown using basic geometric arguments). Thus,

$$\mathbb{P}_T \geq \mathbb{P}_{N_x}^{O\left(\frac{1}{(1-\varepsilon)^2}\right)} \geq \left(\frac{1}{4}\right)^{\frac{1}{2}O\left(\frac{1}{(1-\varepsilon)^2}\right)} \tag{4}$$

Since no other node in T_x is transmitting, we only need to bound the signal received from outside T_x.

To this end, we need the following Claim (which is a restatement of Lemma 4.1 of [24] and can be proven by standard techniques):

Claim B.9. *Assume that for all broadcast regions B_x, $\sum_{y \in B_x} p_y \leq \frac{1}{2}$. Consider a node x. Then the expected power received at node x from nodes not in T_x can be upper bounded by*

$$\frac{1}{8}\frac{\alpha-1}{\alpha-2}3^3 2^{\alpha-2}\frac{(1-\varepsilon)^2}{R^{\alpha}_{(1-\varepsilon)}} \leq \frac{1}{2(4(\beta+4)R_{(1-\varepsilon)})^{\alpha}}$$

for appropriately small $(1-\varepsilon)$.

Then by Markov's inequality, with probability at least $\frac{1}{2}$, the power received from nodes outside of T_x is at most $\frac{1}{(4(\beta+4)R_{(1-\varepsilon)})^{\alpha}}$.

Thus, with probability $\frac{1}{2}\mathbb{P}_T$, LowPower occurs at x, proving the Lemma. $\square$

Lemma B.10. *Consider any node x. Then during any time slot $t \leq 10N_x^2$,*

$$\sum_{y \in B_x} p_y \leq \frac{1}{2} \tag{5}$$

with probability at least $1/2$.

Proof. For contradiction, we will upper bound the probability that Eqn. 5 is violated for the first time at any given time t, after which we will union bound over all $t \leq 10N_x^2 \leq 10\tilde{N}_x{}^2$.

Let $\mathcal{T}$ be the interval (time period) $\{t - \delta\log(\tilde{N}_x/\varepsilon_{ack}) + 1 \ldots t-1\}$. Then we claim,

Claim B.11. *In each time slot in the period $\mathcal{T}$,*

$$\frac{1}{2} \geq \sum_{y \in B_x} p_y \geq \frac{1}{4} \tag{6}$$

Proof. The first inequality is by the assumption that t is the first slot when Eqn. 5 is violated. The second is because probabilities (at most) double once every $\delta \log(\tilde{N}_x/\varepsilon_{ack})$ slots (by the description of the algorithm). □

We now show that Eqn. 6 is not possible. To that end, we show that in the $\delta \log(\tilde{N}_x/\varepsilon_{ack})$ interval preceding t, a FALLBACK will occur with high probability:

Claim B.12. *With probability $1 - \frac{1}{N_x^8}$, each node $z \in B_x$ will* FALLBACK *once in the period $\mathcal{T}$.*

Proof. Fix any $z \in B_x$. By the algorithm

$$p_z \leq \frac{1}{16} \tag{7}$$

Thus, at any time slot,

$$\mathbb{P}(z \text{ does not transmit}) \geq \frac{15}{16} \tag{8}$$

Now, combining Eqn. 7 and Eqn. 6, and defining $B = B_x \setminus \{z\}$,

$$\sum_{y \in B} p_y \geq \frac{3}{16} \tag{9}$$

For $y \in B_x$ define $\textsc{Success}_y$ to be the event that y transmits and LOWPOWER occurs for y. By Lemma B.5, $\textsc{Success}_y$ implies that z will receive the message from y. Thus, the probability of z receiving a message from some node in B in a given round is at least $\frac{15}{16}\mathbb{P}(\bigcup\limits_{y \in B} \textsc{Success}_y)$.

We claim that for any $y \neq w$ (both in B), the events $\textsc{Success}_y$ and $\textsc{Success}_w$ are disjoint. This is implicit in Lemma B.5, since $\textsc{Success}_y$ means that w cannot be transmitting and vice-versa. Thus, the probability of z receiving a message from some node in B is at least:

$$\frac{15}{16}\mathbb{P}(\bigcup_{y \in B} \textsc{Success}_y) = \frac{15}{16}\sum_{y \in B} \mathbb{P}(\textsc{Success}_y)$$
$$\geq \frac{15}{16}\sum_{y \in B} p_y \frac{1}{2}\left(\frac{1}{4}\right)^{\frac{1}{2}O\left(\frac{1}{(1-\varepsilon)^2}\right)} \geq \frac{15}{32}\left(\frac{1}{4}\right)^{\frac{1}{2}O\left(\frac{1}{(1-\varepsilon)^2}\right)} \frac{3}{16}\ ,$$

where we use Lemma B.8 for the first inequality and Eqn. 9 for the last.

Setting $\delta \geq \frac{10}{\frac{15}{32}\left(\frac{1}{4}\right)^{\frac{1}{2}O(\frac{1}{(1-\varepsilon)^2})}\frac{3}{16}}$ and using the Chernoff bound, we can show that z will receive $> 8\log(2\tilde{N}_x/\varepsilon_{ack})$ messages in $\mathcal{T}$ with probability $1 - \frac{1}{\tilde{N}_x}$, thus triggering the FALLBACK. □

Now we show that the above claim implies that Eqn. 6 is not possible.

Claim B.13. *There exists a time slot in $\mathcal{T}$ such that*
$\sum_{y \in B_x} p_y < \frac{1}{4}$.

Proof. For any $y \in B_x$, let p_y^1 be the value of p_y in the first slot of $\mathcal{T}$. Let p_y^f be the value of p_y in the slot when FALLBACK happened for y. Since probabilities can at most double during $\mathcal{T}$,

$$\sum_{y \in B_x} p_y^f \leq 2 \sum_{y \in B_x} p_y^1 \leq 1 \ , \tag{10}$$

the last inequality using the fact that $\sum_{y \in B_x} p_y^1 \leq \frac{1}{2}$ (Eqn. 6).

Now by lines 4 and 7 of the algorithm, in the slot after FALLBACK, $p_y = \max\{\frac{1}{128\tilde{N}_x}, \frac{p_y^f}{32}\} \leq \frac{1}{128\tilde{N}_x} + \frac{p_y^f}{32}$. Since probabilities at most double during $\mathcal{T}$, the value of p_y at the final slot of $\mathcal{T}$ is at most $\frac{1}{64\tilde{N}_x} + \frac{p_y^f}{16}$. Summing over all y, during the final slot of $\mathcal{T}$,

$$\sum_{y \in B_x} p_y \leq \frac{N_x}{32\tilde{N}_x} + \sum_{y \in B_x} \frac{p_y^f}{8} \leq \frac{1}{32} + \frac{1}{8} < \frac{1}{4}$$

contradicting Eqn. 6. We used Eqn. 10 in the second inequality. ◻

The proof of the Claim is completed by union bounding over time slots $t \leq 10N_x^2 \leq 10\tilde{N}_x{}^2$. ◻

Now we prove that nodes stop running the algorithm by a certain time.

Lemma B.14. *Each node x stops executing within $O(N_x \log(\tilde{N}_x) + \log^2(\tilde{N}_x) + \log(\tilde{N}_x/\varepsilon_{approg}))$ slots, with probability at least $1 - \varepsilon_{ack}/2$.*

Proof. Fix x. We derive four claims that together imply the lemma.

Claim B.15. *The number of slots for which $p_x \geq \frac{1}{32}$ is $O(\log(\tilde{N}_x/\varepsilon_{ack}))$.*

Proof. This is ensured by the halting condition in line 14. ◻

Assume that x experienced k FALLBACKs. Consider the times $t_x(1), t_x(2) \ldots t_x(k)$ when a FALLBACK happened for x. Now,

Claim B.16. *$t_x(1) = O(\log(\tilde{N}_x)\log(\tilde{N}_x/\varepsilon_{ack}))$. Also, there are $O(\log(\tilde{N}_x)\log(\tilde{N}_x/\varepsilon_{ack}))$ slots after $t_x(k)$.*

Proof. The two claims are very similar. Let us prove the latter one. Since FALLBACK does not occur after $t_x(k)$, the probability p_x of each node doubles every $\delta \log(\tilde{N}_x/\varepsilon_{ack})$ slots. Since the minimum probability is $\Omega(\tilde{N}_x)$, by $O(\log(\tilde{N}_x)\log(\tilde{N}_x/\varepsilon_{ack}))$ slots, the probability will reach $\frac{1}{32}$. Once this happens, the algorithm terminates in $O(\log(\tilde{N}_x/\varepsilon_{ack}))$ additional slots, by Claim B.15. ◻

Given the above claim it suffices to bound $t_x(k) - t_x(1)$. By Claim B.15 we can also restrict ourselves to slots for which $p_x < \frac{1}{32}$. For these slots, line 7 does not need the min clause, i.e., $p_y \leftarrow 2p_y$ each time line 7 is executed.

Define b_i such that $p_x = \frac{1}{2^{b_i}}$ at time $t_x(i)$. Note that if $\tilde{N}_x$ is a power of 2, b_i is always an integer (the case of other values of $\tilde{N}_x$ can be easily managed).

We can characterize the running time between two FALLBACKs as follows.

Claim B.17. *$t_x(i+1) - t_x(i) \le (b_i - b_{i+1} + 5)\delta \log(\tilde{N}_x/\varepsilon_{ack})$, for all $i = 1, 2 \ldots k-1$.*

Proof. During slots in $[t_x(i), t_x(i{+}1))$, p_x doubles every $\delta \log(\tilde{N}_x/\varepsilon_{ack})$ slots (by the description of the algorithm and the fact that $p_x < \frac{1}{32}$). Let b be such that $p_x = \frac{1}{2^b}$ at time $t_x(i+1) - 1$. Then,

$$\frac{1}{2^b} = \frac{2^{\left\lfloor \frac{t_x(i+1) - t_x(i)}{\delta \log(\tilde{N}_x/\varepsilon_{ack})} \right\rfloor}}{2^{b_i}}$$
$$\Rightarrow b_i - b = \left\lfloor \frac{t_x(i+1) - t_x(i)}{\delta \log(\tilde{N}_x/\varepsilon_{ack})} \right\rfloor$$

By lines 7 and 4 of the algorithm, $b_{i+1} \le b + 4$, and thus,

$$b_i - b_{i+1} + 4 \ge \left\lfloor \frac{t_x(i+1) - t_x(i)}{\delta \log(\tilde{N}_x/\varepsilon_{ack})} \right\rfloor$$
$$\Rightarrow b_i - b_{i+1} + 5 \ge \frac{t_x(i+1) - t_x(i)}{\delta \log(\tilde{N}_x/\varepsilon_{ack})} \ ,$$

completing the proof of the Lemma. □

Thus, the running time $t_x(k) - t_x(1)$ can be bounded by:

$$\begin{aligned} & t_x(k) - t_x(1) \\ & = (t_x(k) - t_x(k-1)) + (t_x(k-1) - t_x(k-2)) \\ & \qquad \ldots + (t_x(2) - t_x(1)) \\ & \le ((b_{k-1} - b_k + 5) + (b_{k-2} - b_{k-1} + 5) \\ & \qquad \ldots + (b_1 - b_2 + 5))\delta \log(\tilde{N}_x/\varepsilon_{ack}) \\ & = (b_1 - b_k + 5k)\delta \log(\tilde{N}_x/\varepsilon_{ack}) \\ & = O(\log(\tilde{N}_x) \log(\tilde{N}_x/\varepsilon_{ack}) + k \log(\tilde{N}_x/\varepsilon_{ack})) \ , \end{aligned} \tag{11}$$

where we use Claim B.17, the non-negativity of b_k and the fact that $b_i = O(\log \tilde{N}_x)$ (as $p_x = \Omega(\frac{1}{\tilde{N}_x})$).

To complete the proof of the Lemma, we need a bound on k:

Claim B.18. *With probability $1 - \varepsilon_{ack}/2$, each node transmits at least $4\gamma' \log(\tilde{N}_x/\varepsilon_{ack})$ times, and at most $16\gamma' \log(\tilde{N}_x/\varepsilon_{ack})$ times.*

Proof. By the description of the algorithm, when the node stops, its total transmission probability is $\gamma' \log(\tilde{N}_x/\varepsilon_{ack})$. By the standard Chernoff bound, the actual number of transmissions is very close to this number, with probability at least $1 - \varepsilon_{ack}/(2N_x^8)$, which is at least $1 - \varepsilon_{ack}/2$. □

Claim B.19. *$k = O(N_x)$ with probability at least $1 - \varepsilon_{ack}/2$.*

Proof. The total number of possible transmissions that x could possibly hear is upper bounded by $O(N_x \log(\tilde{N}_x/\varepsilon_{ack}))$, with probability at least $1-\varepsilon_{ack}/(\text{poly } N_x)$ (due to a Chernoff bound). (However, we only need probability at least $1-\varepsilon_{ack}/4$ for our purposes.) This is because each node transmits $O(\log(\tilde{N}_x/\varepsilon_{ack}))$ times, with probability at least $1-\varepsilon_{ack}/4$) (by Claim B.18) and a node can only hear messages from nodes in T_x (by the definition of T_x). But nodes only FallBack once for every $8\log(\tilde{N}_x/\varepsilon_{ack})$ messages received (by the condition immediately preceding line 20). The claim is proven with probability guarantee $(1-\varepsilon_{ack}/4)(1-\varepsilon_{ack}/4) \geq (1-\varepsilon_{ack}/2)$. $\square$

Applying the above claim to Eqn. 11,

$$\begin{aligned} t_x(k) - t_x(1) &\leq O(\log(\tilde{N}_x)\log(\tilde{N}_x/\varepsilon_{ack}) + k\log(\tilde{N}_x/\varepsilon_{ack})) \\ &= O(N_x \log(\tilde{N}_x/\varepsilon_{ack}) + \log(\tilde{N}_x)\log(\tilde{N}_x/\varepsilon_{ack})), \end{aligned}$$

completing the argument. $\square$

The final piece of the puzzle is to show that for each node, a successful local broadcast happens with probability at least $1-\varepsilon_{ack}/2$ during one of its $\Theta(\gamma' \log(\tilde{N}_x/\varepsilon_{ack}))$ transmissions.

Lemma B.20. *By the time a node halts, it has successfully locally broadcast a message, with probability at least* $1-\varepsilon_{ack}/2$.

Proof. The expected number of transmission made by a node is $\gamma' \log(\tilde{N}_x/\varepsilon_{ack})$ (by the algorithm). By Lemma B.8 (which can be applied, as Lemma B.8's prerequisites are met each time with probability 1/2 due to Lemma B.10) and Lemma B.5, during each such transmission, local broadcast succeeds with probability $\frac{1}{2}\left(\frac{1}{4}\right)^{\frac{1}{2}O(\frac{1}{(1-\varepsilon)^2})}$, at least. Thus, the expected number of successful local broadcasts is $(1-1/2)\cdot\frac{1}{2}\left(\frac{1}{4}\right)^{\frac{1}{2}O(\frac{1}{(1-\varepsilon)^2})}\gamma' \log(\tilde{N}_x/\varepsilon_{ack})$. Setting γ' to a high enough constant, and using Chernoff bounds, with probability at least $1-\varepsilon_{ack}/2$, a successful local broadcast happens at least once. $\square$

Lemmas B.14 and B.20 together imply Thm. B.3 with probability guarantee $(1-\varepsilon_{ack}/2)(1-\varepsilon_{ack}/2) \geq (1-\varepsilon_{ack})$.

C Useful Lemmas and Proofs from [13] Adapted to our Needs

We restate two lemmas and proofs from [29] adapted to our needs for completeness. This is done only for the convenience of the reader with the goal of making it simpler to verify our claim. Compared to the adapted proofs in the main-body of the paper, the proofs presented here have only minor modifications and are adapted to our notation.

Lemma C.1 (Version of Lemma 4.4 of [13])**.** *Given node $i \in N_{G_{1-2\varepsilon}}(S_1)$ and assume Properties 1 and 2 of Definition 10.8 of a successful epoch at point i are satisfied. Then for any $\phi \in \{1,\ldots,\Phi\}$, the minimum distance between any two nodes in $S_{\phi,i}$ is at least* $d_\phi \geq 2^{\phi-1}\cdot d_{\min}$.

Proof. For completeness and clarity we restate the full proof of [13] and extend it to our setting. We prove the lemma by induction on ϕ. By definition of $d_{\min}$, it is $d_1 \geq 2^0 \cdot d_{\min} = d_{\min}$. By the definition of γ'-close approximation of $H_p^\mu[S]$ and as we assume that Properties 1 and 2 of Definition 10.8 of a successful epoch at point i are satisfied, we can apply Lemma 10.14 and conclude that $H_p^\mu[S_\phi]|_{S_{\phi,i}}$ contains edges between all pairs of nodes $S_{\phi,i}$ at distance $d(u,v) \leq 2 \cdot d_\phi$. As $S_{\phi+1,i}$ is $(\phi+1,i)$-locally maximal in $\tilde{\tilde{H}}_p^\mu[S_\phi]|_{S_{\phi,i}}$, nodes in $S_{\phi+1,i}$ are at distance more than $2 \cdot d_\phi$. Using the induction hypothesis, it is $d_{\phi+1} > 2 \cdot d_\phi \geq 2^\phi \cdot d_{\min}$. □

Lemma C.2 (Version of Lemma 4.5. of [13])**.** *Assume Property 2 of the For all $p \in (0, 1/2]$, there is a $\hat{Q}, \gamma = \Theta(1)$, such that for all $Q \geq \hat{Q}$ the following holds. Consider a round r in phase ϕ where each node in S_ϕ transmits a bcast-message with probability p/Q (Line 11). Let $i \in N_{G_{1-2\varepsilon}}(S_1)$ and let $u_\phi \in S_\phi \setminus \{v\}$ be the closest node to v in S_ϕ. Assume Property 1 of Definition 10.8 of a successful epoch at point i are satisfied. Let d_{u_ϕ} be the distance between u_ϕ and its farthest neighbor in $\tilde{\tilde{H}}_p^\mu[S_\phi]$. If $d(u_\phi, v) \leq (1+\varepsilon)R_{1-2\varepsilon}$ and $d_{u_\phi} \geq \gamma Q^{-1/\alpha} \cdot d(u_\phi, v)$, node v receives a bcast-message from u_ϕ in round r with probability $\Theta(1/Q)$.*

Proof. For completeness and we restate the full proof of [13] and extend it based on the ideas summarized in the main-body of our paper. The lemma states under what conditions in round r of block 2 in phase ϕ a node $v \in N(S)\backslash S$ can receive the message. The roadmap for this proof is to show that if u is able to communicate with probability $(1-\varepsilon)\mu$ with its farthest neighbor u' in some round r' of block 1 in phase ϕ, using the broadcast probability p, then u must also be able to reach v with probability $\Theta(1/Q)$ in round r of block 2, in which it transmits with probability p/Q. We start with some notations and continue with a connection between the interference at u and at v. We then analyze the interference at u created in a ball of radius $2d_u$ around u, as well as the remaining interference coming from outside that ball. Finally, we transfer all the knowledge we gained for round r' to round r to conclude the proof.

For a node $w \in V$, let $I_{S_\phi \cup W}(w) = \sum_{x \in S_\phi \cup W} \frac{P}{d(x,w)^\alpha}$, i.e., the amount of interference at node w if all nodes of $S_\phi \cup W$ transmit. For round r', the random variable $X_x^p(w)$ denotes the actual interference at node w coming from a node $x \in S$ (the superscript ^{p} indicates the broadcasting probability of nodes in round r'). The total interference at node w is thus $X^p(w) := \sum_{x \in S_\phi \cup W} X_x^p(w)$. If we only want to look at the interference stemming from nodes within a subset $A \subseteq S_\phi$, we use $I_A(w)$ and $X_A^p(w)$ respectively. For round r, in which nodes use the broadcasting probability p/Q, we use the superscript $^{p/Q}$. Finally, for a set $A \subseteq S_\phi$, we define $\bar{A} := S_\phi \backslash A$.

For any $w \in S_\phi$, the triangle inequality implies that $d(u,w) \leq d(u,v) + d(v,w) \leq 2d(v,w)$. By comparing $I_{S'}(u)$ and $I_{S'}(v)$ for an arbitrary set $S' \subseteq S_\phi$ we obtain the following observation:

$$I_{S'}(u) \geq 2^{-\alpha} I_{S'}(v). \tag{12}$$

Let u' be the farthest neighbor of node u in $\tilde{H}_p^\mu[S_\phi]$. Because $\tilde{H}_p^\mu[S_\phi]$ is an ε-close approximation of $H_p^\mu[S_\phi]$, we know that $\tilde{H}_p^\mu[S_\phi]$ and that this is a subgraph of $H_p^{(1-\varepsilon)\mu}[S_\phi]$. Therefore [13] now argues that in round r', u receives a message from u_ϕ with probability at least $(1-\varepsilon)\mu$. In our case, we can only claim that $\tilde{\tilde{H}}_p^\mu[S_\phi]$ is a subgraph of $H_p^{(1-\varepsilon)\mu}[S_\phi]$ in a certain area around u. It turns out that this is sufficient, as we argue below.

Let $A \subseteq S_\phi$ be the set of nodes at distance at most $2d_u$ from u. Note that $d(u, u_\phi) = d_u$ and therefore both u and u_ϕ are in A. In round r', if more than $2^\alpha/\beta = \mathcal{O}(1)$ nodes $u' \in A$ transmit, then node u cannot receive a message from u_ϕ. Since node u receives a message from u_ϕ with probability at least $(1-\varepsilon)\mu$ in round r', we can conclude that fewer than $2^\alpha/\beta$ nodes transmit with at least the same probability.

We show that this disc of radius $2d_u$ around u is covered by a $\mathcal{O}(1)$-neighborhood of u_ϕ in $\tilde{\tilde{H}}_p^\mu[S_\phi]$. For sake of contradiction assume Lemma 10.16 is not true while Property 1 of Definition 10.8 of a successful epoch at point i is satisfied. Then the communication link between u_ϕ and its furthest neighbor in $H_p^\mu[S_\phi]$ could not be μ-reliable, as there are $\omega(1)$ nodes within distance $2d_{u_\phi}$ that are sending with probability p each. We now bound the interference from nodes outside of A. The authors of [13] prove that $I_{\bar{A}}(u) \le c \cdot \frac{P}{p\beta d_v^\alpha}$, where c is a constant. However, compared to [13] we need to take interference from nodes W into account, as already pointed out in the proof of Lemma 10.14, and we modify their proof to derive $I_{\bar{A}\cup W}(u) \le c' \cdot \frac{P}{p\beta d_v^\alpha}$ for some constant c'. Using the fact that node u receives a message from node u_ϕ with constant probability at least $(1-\varepsilon)\mu$ allows us to upper bound $I_{\bar{A}\cup W}(u)$ and by (12) also $I_{\bar{A}\cup W}(v)$. For node u to be able to receive a message from u_ϕ, two things must hold:
(a) $\frac{P}{d_u^\alpha(N + X^p_{\bar{A}\cup W}(u))} \ge \frac{P}{d_u^\alpha(N+X^p(u))} \ge \beta$ and
(b) u_ϕ transmits and u listens (event R^{u,u_ϕ}).

Due to Lemma 10.3 we know that $I_W(u) \le \left(\frac{\varepsilon_{approg}}{\Lambda}\right)^{\Theta(1)}$. This implies, that we can transform (a) to $\frac{P}{d_u^\alpha(N+X^p_{\bar{A}}(u))} \le c' \cdot \frac{P}{p\beta d_v^\alpha}$ for some c', when we choose the exponent hidden in Θ-notation to match the choice of constant c'. Thus we have for $X^p_{\bar{A}}(u)$ that

$$(1-\varepsilon)\mu \le \mathbb{P}R^{u,u_\phi} \cdot \mathbb{P}\left(X^p_{\bar{A}}(u) \le \frac{P}{\beta d_u^\alpha} - N\right) \le p(1-p)\cdot \mathbb{P}\left(X^p_{\bar{A}}(u) \le \frac{P}{\beta d_u^\alpha}\right). \tag{13}$$

Using Lemma B.1, we can therefore bound $X^p_{\bar{A}}(u)$ as

$$\mathbb{P}\left(X^p_{\bar{A}}(u) \le \frac{\mathbb{E}[X^p_{\bar{A}}(u)]}{2}\right) = \mathbb{P}\left(X^p_{\bar{A}}(u) \le \frac{pI_{\bar{A}}(u)}{2}\right) \le e^{-\frac{p2^\alpha d_u^\alpha}{8p}\cdot I_{\bar{A}}(u)}. \tag{14}$$

For the sake of contradiction, assume that $I_{\bar{A}}(u) > c \cdot \frac{P}{p\beta d_u^\alpha}$ for $c = \max\{2, \frac{16\beta}{2^\alpha} \cdot \ln\frac{p(1-p)}{(1-\varepsilon)\mu}\}$. Combining (13) and (14), we obtain

$$\frac{(1-\varepsilon)\mu}{p(1-p)} \overset{(2)}{\le} \mathbb{P}\left(X^p_{\bar{A}}(u) \le \frac{P}{\beta d_u^\alpha}\right) \le \mathbb{P}\left(X^p_{\bar{A}}(u) \le \frac{cP}{2\beta d_u^\alpha}\right) < \mathbb{P}\left(X^p_{\bar{A}}(u) \le \frac{pI_{\bar{A}}(u)}{2}\right) \le e^{-\frac{2^\alpha c}{16\beta}},$$

which is a contradiction to the definition of c. We therefore have $I_{\bar{A}}(u) \le c \cdot \frac{P}{p\beta d_u^\alpha}$ and $I_{\bar{A}\cup W}(u) \le c' \cdot \frac{P}{p\beta d_u^\alpha}$.

We now have all tools to show that v receives a message from u in round r, with broadcasting probabilities p/Q. From the fact that the link $\{u, u_\phi\} \in E[\tilde{H}_p^\mu[S_\phi]]$ is reliable, we have seen that with probability at least $(1-\varepsilon)\mu$ fewer than $\frac{2^\alpha}{\beta}$ nodes in A send in round r'. But then in round r with the same probability no more than $\frac{2^\alpha}{\beta Q}$ send within A. Markov's inequality shows that $\mathbb{P}\left(X^{p/Q}_{\bar{A}\cup W}(v) < 2\frac{p}{Q}I_{\bar{A}\cup W}(v)\right) \ge 1/2$. Finally, u sends with probability

p/Q. All those events are independent, thus all of them happen with probability at least $\frac{(1-\varepsilon)\mu p}{2Q} = \Theta(1/Q)$. Let us assume that this is the case. To see that v indeed gets u's message under those conditions, we check whether $SINR(u,v,I) = \frac{Pd(u,v)^{-\alpha}}{N+X^{p/Q}_{\tilde{A}\cup W}(v)+X^{p/Q}_{A}} \geq \beta$:

$$\beta d(u,v)^{\alpha}(N + X^{p/Q}_{\tilde{A}\cup W}(v) + X^{p/Q}_{A})$$

$$\begin{aligned}
&\overset{(*)}{\leq} && \beta d(u,v)^{\alpha} N + 2^{\alpha+1} c_{\beta} P \frac{d(u,v)^{\alpha}}{d_u^{\alpha}} + \beta \sum_{w\in A, w \text{ sends}} P \frac{d(u,v)^{\alpha}}{Q d(w,v)^{\alpha}} \\
&\overset{d(u,v)^{\alpha}\leq Q\frac{d_u^{\alpha}}{\gamma^{\alpha}}}{\leq} && \left(1+\frac{\rho}{2}\right)^{\alpha} r_s^{\alpha} N\beta + \frac{2^{\alpha+1}c_{\beta}}{\gamma^{\alpha}} P + \frac{2^{\alpha}}{Q} P \\
&\overset{(1+\rho)^{\alpha}\geq\left(1+\frac{\rho}{2}\right)^{\alpha}+\alpha\frac{\rho}{2}}{\leq} && \left(1 - \frac{\alpha\rho}{2(1+\rho)^{\alpha}}\right)(1+\rho)^{\alpha} r_s^{\alpha} N\beta + \frac{2^{\alpha+1}c_{\beta}}{\gamma^{\alpha}} P + \frac{2^{\alpha}}{\hat{Q}} P \\
&\overset{P=N\beta(1+\rho)^{\alpha}r_s^{\alpha}}{\leq} && P + P\left(\frac{2^{\alpha+1}c_{\beta}}{\gamma^{\alpha}} + \frac{2^{\alpha}}{\hat{Q}} - \frac{\alpha\rho}{2(1+\rho)^{\alpha}}\right) \\
&\overset{(**)}{\leq} && P
\end{aligned}$$

Inequality $(*)$ holds due to the assumption that $X^{p/Q}_{\tilde{A}\cup W}(v) < 2I_{\tilde{A}\cup W}(v)p/Q$ and (13). Inequality $(**)$ holds for properly chosen $\gamma = \Theta(1)$ and $\hat{Q} = \Theta(2^{\alpha}) = \mathcal{O}(\log^{\alpha}_{\max} R_s)$. ☐

Lemma C.3 (Version of Lemma 4.6. of [13]). *Assume Property 2 of the Definition 10.8 of a successful epoch is satisfied. With probability $1 - \varepsilon_{approg}/3$, either u_{ϕ}'s bcast-message reaches i in phase ϕ, or $d(u_{\phi+1}, i) \leq R_{1-\varepsilon}\left(1 + \phi\frac{\varepsilon}{\log\Lambda}\right)$.*

Proof. For completeness and clarity we restate the full proof of [13] and extend it to our setting. Clearly, $d(u_1, i) \leq R_{1-2\varepsilon}$. Let ϕ be any phase. If $d_{u_{\phi}} \geq Q^{-1/\alpha} d(u_{\phi}, i)$, then we can apply Lemma 10.16 and we are done, because u_{ϕ} sends for $\mathcal{O}(Q\log(1/\varepsilon_{approg}))$ rounds in Line 11. To see this, we choose the constant hidden in the $\mathcal{O}$-notation large enough, and derive that with probability $1-(1-(1/\Theta(Q))^{\mathcal{O}(Q\cdot\log(1/\varepsilon_{approg}))} \geq 1-e^{-\log(3/\varepsilon_{approg})} = 1-\varepsilon_{approg}/3$ the bcast-message sent by u_{ϕ} reaches i during the execution of Lines 10–13. Now let this not be the case and let $u_{\phi+1}$ be the closest neighbor to i in $S_{\phi+1}$. Due to the assumption that Property 2 of Definition 10.8 of a successful epoch at point i is satisfied, $S_{\phi+1}$ is (ϕ, i)-locally maximal in $\tilde{\tilde{H}}^{\mu}_{p}[S_{\phi}]$. Using this maximality property of our construction, it is $d(u_{\phi+1}, i) \leq d(u_{\phi}, i) + d_{u_{\phi}}$, and therefore

$$\begin{aligned}
d(u_{\phi+1}, i) &\leq \left(1 + \frac{\gamma'}{Q^{1/\alpha}}\right) d(u_{\phi}, i) \leq R_{1-2\varepsilon}\left(1 + \phi\frac{\varepsilon}{\log\Lambda} + \frac{2\gamma'}{Q^{1/\alpha}}\right) \\
&\leq R_{1-2\varepsilon}\left(1 + \frac{(\phi+1)\varepsilon}{\log\Lambda}\right).
\end{aligned}$$

The last inequality holds for properly chosen $Q = \Theta\left(\log^{\alpha} R_{1-2\varepsilon}\right)$, $Q \geq \hat{Q}$, proving the Lemma. ☐