Theory Comput Syst (2010) 47: 811–836
DOI 10.1007/s00224-010-9271-x

Distributed Approximation of Capacitated Dominating Sets

Fabian Kuhn · Thomas Moscibroda

Published online: 2 June 2010

Abstract We study local, distributed algorithms for the capacitated minimum dominating set (CapMDS) problem, which arises in various distributed network applications. Given a network graph $G = (V, E)$, and a capacity $\mathrm{cap}(v) \in \mathbb{N}$ for each node $v \in V$, the CapMDS problem asks for a subset $S \subseteq V$ of minimal cardinality, such that every network node not in S is covered by at least one neighbor in S, and every node $v \in S$ covers at most $\mathrm{cap}(v)$ of its neighbors. We prove that in general graphs and even with uniform capacities, the problem is inherently *non-local*, i.e., every distributed algorithm achieving a non-trivial approximation ratio must have a time complexity that essentially grows linearly with the network diameter. On the other hand, if for some parameter $\epsilon > 0$, capacities can be violated by a factor of $1 + \epsilon$, CapMDS becomes much more local. Particularly, based on a novel distributed randomized rounding technique, we present a distributed bi-criteria algorithm that achieves an $\mathrm{O}(\log \Delta)$-approximation in time $\mathrm{O}(\log^3 n + \log(n)/\epsilon)$, where n and Δ denote the number of nodes and the maximal degree in G, respectively. Finally, we prove that in geometric network graphs typically arising in wireless settings, the uniform problem can be approximated within a constant factor in logarithmic time, whereas the non-uniform problem remains entirely non-local.

Keywords Capacities · Dominating sets · Distributed approximation · LP relaxation · Locality · Lower bounds

A preliminary version of this paper has appeared in the Proceedings of the 19th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), San Diego, California, USA, June 2007.

F. Kuhn
Faculty of Informatics, University of Lugano, Lugano, Switzerland
e-mail: fabian.kuhn@usi.ch

T. Moscibroda (✉)
Microsoft Research, Redmond, WA, 98052, USA
e-mail: moscitho@microsoft.com

1 Introduction

In large-scale and highly-decentralized networks, the concept of clustering plays an important role in a variety of network coordination tasks. In wireless multi-hop networks, for instance, self-organized clustering of nodes has been proposed and used for facilitating communication between nodes in physical proximity (MAC layer protocols [16]), for enabling efficient routing [32–34]), or to improve localization [7] and energy efficiency [9]. As different as these applications are, the required structures are typically based on key primitives and boil down to classic graph-theoretic objects.

Possibly the most well-studied combinatorial optimization problems in this context is the minimum dominating set (MDS) problem. In this problem, given a graph $G = (V, E)$, the goal is to choose a subset $S \subseteq V$ of minimal cardinality such that every node $V \setminus S$ has at least one neighbor in S. In a sensor network, for instance, if only nodes in S stay awake, each remaining node can go to an energy-saving sleep-mode while still having an active node within its communication range.

Although the MDS problem formulation thus captures an important problem domain, it has the shortcoming that it allows (in fact, favors) solutions in which some nodes cover a large number of neighboring nodes. In principle, applying an MDS algorithm therefore requires that each node can manage *all* its neighbors, as this situation may turn out to be the solution. Considering that a cluster center offers a service to all its "clients", cluster centers can handle only a limited number of covered nodes.

In this paper, we therefore study distributed approximation algorithms for the *capacitated minimum dominating set problem* (CapMDS). In this problem, every network node $v \in V$ can cover only a certain number $\text{cap}(v)$ of neighboring nodes. The task is to select a subset $S \subseteq V$ of dominators and to assign each of the remaining nodes $V \setminus S$ to one of the dominators such that all capacity bounds are satisfied. To the best of our knowledge, this paper presents the first distributed algorithms and lower bounds for the capacitated dominating set problem.

In particular, we are interested in devising *local, distributed algorithms* for the CapMDS problem or proving the impossibility of such algorithms. A local algorithm is a distributed algorithm that bases its decision only on information in its local neighborhood (as opposed to requiring global knowledge). Technically speaking, a distributed algorithm is called *local* if its distributed time complexity is significantly smaller than the network's diameter [18, 25, 29].

We derive a number of results on the distributed complexity and local approximability of the CapMDS problem, both in *general graphs* as well as in *bounded independence graphs* (BIGs). These graphs are a non-geometric generalization of unit disk graphs (UDG) and more accurately model the communication topologies encountered in wireless networks. In general graphs, we prove that even the uniform CapMDS problem cannot be approximated to within a non-trivial approximation ratio by any local algorithm. If small violations of capacities are allowed, much better solutions become possible. Specifically, we present a bi-criteria algorithm that achieves an $\mathrm{O}(\log \Delta)$-approximation in time $\mathrm{O}(\log^3(n) + \log(n)/\epsilon)$ and violates capacity constraints by at most a factor of $1 + \epsilon$ for an arbitrary parameter $\epsilon > 0$. We also prove that any distributed algorithm that violates constraints only by a factor of $1 + \epsilon$ must have a time complexity of at least $\Omega(1/\epsilon \cdot \sqrt{\log n / \log \log n})$ in order to achieve a polylogarithmic approximation ratio. In contrast to this lower bound in general graphs, we show that better solutions are possible for bounded independence graphs. In particular, we present a local, distributed algorithm for the uniform CapMDS problem in such graphs that has constant approximation ratio and polylogarithmic time complexity.

In a broader context, these results shed new light on the *local approximability* of an important capacitated network problem. Modern distributed systems such as peer-to-peer networks, wireless sensor networks, or the Internet have grown so large that in many cases, global requirements, solutions, or equilibria can only be achieved by *local algorithms*. Inevitably, if the number of communication rounds is smaller than the network's diameter, each node can base its decision only on partial, in fact *local knowledge*. This observation evokes the key question of *what can and what cannot be computed locally?* [18, 25, 29].

In recent years, the distributed computing community has therefore been interested in characterizing the local and *distributed approximability* of combinatorial optimization problems, e.g. [6, 10, 20, 24]. Clearly, there exists a trade-off between the amount of local knowledge and the time-complexity of the distributed algorithm on the one hand, and the achievable global approximation ratio on the other hand. Our lower bounds and distributed approximation algorithms characterize this trade-off, thereby capturing the inherent locality of the CapMDS problem.

The remainder of this paper is organized as follows. An overview of relevant previous work is given in Sect. 2. The different network and communication models are introduced in Sect. 3. The main technical contributions are then presented in Sects. 4 and 5 in which we give upper and lower bounds on the distributed approximability of the CapMDS problem in general graphs and bounded independence graphs, respectively. Finally, Sect. 6 relates our results to known results for the basic MDS problem and concludes the paper.

2 Related Work

Because of its practical importance in different areas of networking, and in the wake of a genuine interest in the distributed approximability of combinatorial optimization problems, there has recently been a lot of work on distributed algorithms for finding minimum dominating sets.

MDS–General Graphs: Since any maximal independent set (MIS) constitutes a dominating set in a graph, the earliest distributed algorithms for the MDS problem are in fact the classic distributed MIS algorithms in [1, 27]. These algorithms compute a MIS in time $O(\log n)$, but they do not provide any non-trivial approximation guarantees in general since the size of a MIS can be by a factor of n larger than the size of an optimal MDS in general graphs. Several subsequently proposed distributed MDS algorithms also provide either no guarantees on the time complexity or on the approximation guarantees [23, 34]. The first distributed algorithm with both bounded time complexity and approximation guarantees was given by Jia et al. [17]. This randomized algorithm achieves an approximation ratio of $O(\log \Delta)$ in time $O(\log n \log \Delta)$, where n is the number of nodes and Δ is the largest degree in the network. Later these bounds were improved by Kuhn et al. [20, 22]. In particular, the work in [20] presents a local algorithm that, for any constant k, achieves an approximation factor of $O(\Delta^{1/\sqrt{k}} \log \Delta)$ in time $O(k)$. On the other hand, it was shown by Kuhn et al. in [18] that this is close to optimal since there exists no distributed algorithm with running time k that can achieve a better approximation ratio than $\Omega(\Delta^{1/k}/k)$.

MDS–Geometric Graph Classes: For the unit disk graph and generalizations thereof, a multiplicity of algorithms have been proposed. Again, most of the earlier proposals have the property that in the worst case, they either have linear time complexity (i.e., are not local) [2, 32] or achieve no non-trivial approximation guarantees [8, 34]. There has recently been a series of work trying to establish the exact power of local algorithms in unit disk graphs, and interestingly, it turns out that this power depends on the amount of additional information given to the nodes. Three models have been studied:

- If nodes know their *location* or coordinates, a straightforward single-round algorithm can compute a constant approximation to the MDS problem [19].
- If nodes have only *distance information* to their neighbors, it was shown by Gao et al. that a constant approximation can be computed in time $O(\log \log n)$ [12]. This was later improved to $O(\log^* n)$ in [19].[1]
- If nodes do not have *any such extra information*, deterministic and randomized local, constant-approximation algorithms with time complexities of $O(\log \Delta \log^* n)$ and $O(\log \log n \log^* n)$ were proposed in [21] and [14], respectively. Most recently, it was shown in [31] that even without distance information, a constant approximation can be computed in time $O(\log^* n)$. Furthermore, this was shown to be optimal as shown by a matching lower bound of $\Omega(\log^* n)$ in [24].

The problem has also been studied in harsher radio network models that allow for possible collisions of (wireless) messages. The work in [11] proposed an algorithm that computes a constant approximation to the connected dominating set problem in polylogarithmic time. In an even harsher model of computation, [28] presents an algorithm with time complexity $O(\log^2 n)$ for computing a maximal independent set in unit disk graphs.

Capacitated MDS: In contrast to the regular dominating set problem, there exists no previous work on the distributed computation of capacitated dominating sets. *Centralized approximation algorithms* for the problem were given by Bar-Ilan et al. in [4] and [5], respectively. More specifically, the work in [4] presents approximation algorithms for a variety of NP-hard capacitated network center allocation problems. For the capacitated dominating set problem with uniform capacities, they give an elegant greedy algorithm, which achieves an optimal approximation ratio of $\ln n$, unless $NP = DTIME(n^{O(\log \log n)})$. The only paper that studies the distributed approximability of a capacitated covering problem is the work of Grandoni et al. [15] on the *capacitated vertex cover problem*.

[1] The function $\log^* n$ is defined as the number of times we have to take the logarithm before the value decreases below 2. For instance, if $\log \log n > 2$ and $\log \log \log n < 2$, then $\log^* n = 3$.

3 Model and Definitions

We model the network as a graph $G = (V, E)$, where vertices and edges represent network nodes and communication links, respectively. Every node has a unique identifier $\mathrm{ID}(v)$ (for instance its IP-address) and, at the outset of the algorithm, has no global knowledge about the network. For the sake of simplicity, we consider the standard *synchronous message passing model* in which time is divided into *communication rounds*. In each round, a node can send a message to all its neighbors. However, note that at the cost of higher message complexity, our algorithms can also be deployed in asynchronous settings without deteriorating their time complexity using the notion of synchronizers [3]. As for notation, $d(u, v)$ denotes the shortest hop-distance between two nodes u and v. For any $r \geq 0$, and $v \in V$, we define the (closed) r-neighborhood $\Gamma_r(v) := \{u \in V \mid d(u, v) \leq r\}$. As a special case, the 1-neighborhood $\Gamma_1(v)$ consists of v and all its direct neighbors and is abbreviated as $\Gamma(v)$.

In a graph $G = (V, E)$, a *dominating set* $S \subseteq V$ is a subset of the nodes such that every node is either in S or has at least one neighbor in S. The *Minimum Dominating Set Problem* asks for a dominating set of minimal cardinality and is known to be NP-hard [13]. In the *capacitated minimum dominating set problem* (CapMDS), each node $v \in V$ has a capacity that places an upper bound on the number of neighboring nodes that v can cover.

Definition 3.1 Consider a graph $G = (V, E)$. For every node $v \in V$, let $\mathrm{cap}(v) \geq 1$ be the capacity of v. A capacitated dominating set (CapDS) is a subset $S \subseteq V$ and a mapping $\phi : V \to S$, such that $\phi(v) \in \Gamma(v)$ for all $v \in V$ and $|\{u|\phi(u) = v\}| \leq \mathrm{cap}(v)$ for all $v \in S$ holds. The CapMDS problem asks for a CapDS with a set S of minimum cardinality.

The CapMDS problem can be formulated as the following integer linear program $\mathrm{ILP}_{\mathrm{CapDS}}$:

$$\min \sum_{v_i \in V} x_i,$$

$$\sum_{v_j \in \Gamma(v_i)} y_{ji} \geq 1, \quad \forall v_i \in V, \tag{1}$$

$$\sum_{v_j \in \Gamma(v_i)} y_{ij} \leq x_i \cdot \mathrm{cap}(v_i), \quad \forall v_i \in V, \tag{2}$$

$$y_{ij} \leq x_i, \quad \forall v_i, v_j \in V, \tag{3}$$

$$x_i, y_{ij} \in \{0, 1\}, \quad \forall v_i, v_j \in V. \tag{4}$$

The communication graphs formed by wireless networks are much more structured than captured by general graphs. In particular, if two nodes are physically too far from each other, there is guaranteed to be no direct communication link. Moreover, if there are many nodes in close physical proximity, not too many of them can be mutually independent.[2] The frequently studied *unit disk graph* captures these intuitions, but tends to be a too optimistic and rigid model. In this paper, we study a more general family of graphs to model wireless networks.

Definition 3.2 (Bounded Independence Graph [21]) A graph G is called *f-independence-bounded* if there is a function $f(r)$ such that every r-neighborhood $\Gamma_r(v)$ of G contains at most $f(r)$ independent (i.e., pairwise non-adjacent) nodes. A graph G has *polynomially bounded independence* if $f(r)$ is a polynomial in r.

An alternative characterization of a BIG is that there exists a constant C such that every 2-hop neighborhood of any node contains at most C pairwise independent nodes. It is clear that for an BIG, $C \leq f(2)$. On the other hand, it is not difficult to see that the cardinality of the largest independent set in a r-hop neighborhood is at most C^r (per induction, because the 2-hop neighborhood of all MIS-nodes in a $(r-1)$-neighborhood cover the entire r-hop neighborhood.). Note that $f(r)$ depends neither on n nor on any other property of G. Hence, for constant r, the number of independent nodes in an r-neighborhood is constant. Further, notice that an f-independence bounded graph is $K_{1,f(1)+1}$-free.

The advantage of the BIG model compared to the unit disk graph model is that it does not imply an explicit geometry and therefore, obstacles or irregular signal propagation are easily captured in the BIG model. For instance, Fig. 1 shows a network topology that can easily be modeled as a BIG, but does not form a unit disk graph. By a standard area argument, it follows that for unit disk graphs

$$f_{UDG}(r) \leq \frac{(r+\frac{1}{2})^2\pi}{\pi/4} = 4\left(r+\frac{1}{2}\right)^2.$$

For general d-dimensional Euclidean spaces, the corresponding independence function is $f(r) \in O(r^d)$ and a constant degree graph with maximum degree Δ has an independence function of $f(r) \in O(\Delta^r)$.

Finally, a *maximal independent set* (MIS) is a subset $R \subseteq V$ in which every node $v \notin R$ has at least one neighbor in R, but any two nodes in R are independent. While

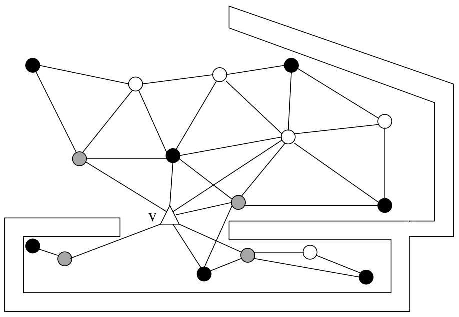

Fig. 1 A bounded independence graph with $f(1) = 4$ and $f(2) = 7$. No node u has more than 4 and 7 mutually independent nodes in its one and two-hop neighborhood, respectively

[2]Two nodes u and v are *independent* if there is no link $(u, v) \in E$. A set of nodes $W \subseteq V$ is mutually independent if there is no link between any two nodes in W.

in a centralized scenario, finding a MIS is completely trivial—pick an arbitrary node and discard all its neighbors from the graph, and repeat this process until there are no more nodes left—, the MIS problem is a core problem in distributed computing because it prototypically captures the fundamental notion of symmetry breaking [25, 27].

4 General Graphs

In this section, we derive upper and lower bounds on the distributed approximability of the capacitated minimum dominating set problem in general graphs.

4.1 Lower Bounds

Consider the capacitated version of the *minimum vertex cover problem*. On a simple ring network, it can easily be seen that this problem is inherently non-local and therefore cannot be efficiently solved by a distributed algorithm: If every node has a capacity of 1 on a ring, all nodes in the ring have to decide on a common direction in order to be able to cover all edges. However, finding such a common direction requires knowledge about the entire ring. In contrast, the *capacitated dominating set problem* intuitively appears to be a much more "local" problem, because every node is capable of covering itself. However, the following theorem proves that the capacitated dominating set is inherently non-local, too, even if capacities are uniform.

Theorem 4.1 *There are graphs G, such that in k communication rounds, every (possibly randomized) distributed algorithm for the minimum capacitated dominating set problem on G has approximation ratios at least*

$$\Omega\left(\frac{n}{k^2}\right) \quad \textit{and} \quad \Omega\left(\frac{\Delta}{k}\right),$$

even if capacities are uniform.

Proof For every $k > 0$, consider a graph I_k as illustrated in Fig. 2. We assume for ease of presentation that k is even, the case where k is odd is analogous. I_k is defined as follows. The node set is partitioned into $k+2$ clusters $C_1, \dots, C_{k+2}$ each containing m nodes. Additionally, there are $k+1$ *connecting nodes* $v_1, \dots, v_{k+1}$. There is an edge between a connecting node v_i and every node in C_i and C_{i+1}. Finally, there is a designated connecting node v_0 that has a link to either all nodes in C_1 or all nodes in C_{k+2}. Let the capacity of all nodes $v \in V$ be $\text{cap}(v) = m+1$.

Let v^* denote the connecting node $v_{k/2+1}$ in the middle of the graph. After communicating for k rounds, every node has only knowledge about its k-hop neighborhood. By the definition of I_k, neither v^* nor any of its neighbors knows the location of v_0. Therefore, the decision of which nodes v^* covers cannot depend on the location of v_0.

Consider the nodes that are covered by v^*. Because $\text{cap}(v^*) = m+1$, at most $m/2$ nodes are covered by v^* in either $C_{k/2+1}$ or $C_{k/2+2}$. Without loss of generality,

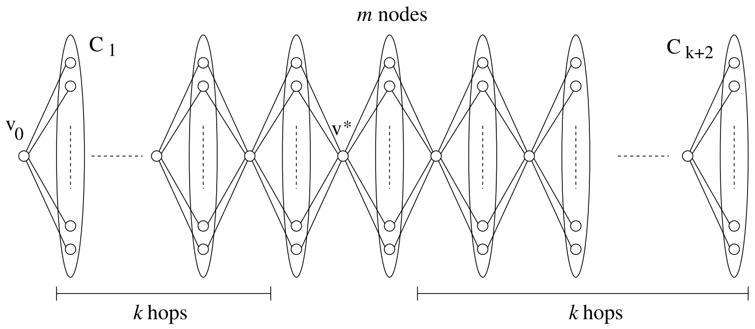

Fig. 2 The structure of lower-bound graph I_k

assume that v^* covers at most $m/2$ nodes in $C_{k/2+2}$. If v_0 is connected to C_1 as in Fig. 2, there are at least $(k/2+1)m$ nodes to the right of v^* that must be covered, but there are only $k/2$ connecting nodes v_j for $j > k/2+1$, each of which can cover at most $m+1$ nodes. The total number of nodes to the right of v^* that can be covered by connecting nodes is therefore at most $k/2 \cdot (m+1) + m/2$. Therefore, at least $(m-k)/2$ nodes in the clusters to the right of v^* must cover themselves and hence, $ALG \geq (m-k)/2$. The optimal solution can cover all nodes using only connecting nodes. Because of $n = (m+1)(k+2)$, the approximation ratio α of every k-local distributed algorithm is therefore at best

$$\alpha \geq \frac{(m-k)/2}{k+2} = \frac{\frac{n}{k+2} - k - 1}{2(k+2)} \in \Omega\left(\frac{n}{k^2}\right).$$

The second bound follows due to $\Delta = 2m$. □

Theorem 4.1 essentially thwarts all our hopes of devising a local approximation algorithm with non-trivial guarantees in general graphs. In the sequel, we therefore focus on the question whether there exist substantially better solutions once we allow small violations of the capacity constraints. As it turns out, even in this case, no constant-time distributed algorithm can approximate CapMDS within a polylogarithmic approximation ratio.

In order to prove this result, we present a *locality-preserving reduction* [18] from CapMDS with $(1+\epsilon)$-violations to the *fractional minimum vertex cover* (MVC) problem. In this problem, given a graph $G=(V,E)$, we need to assign a fractional value $0 \leq a_i \leq 1$ to each node $v_i \in V$ such that each edge is *covered*, i.e., for each $(u,v) \in E$, it holds that $a_u + a_v \geq 1$. The goal is to minimize $\sum_{v \in V} a_v$.

Our locality-preserving reduction shows that a local distributed algorithm for solving CapMDS with $(1+\epsilon)$-violations can be used to devise a local distributed approximation algorithm for MVC. Since it has been proven that no such distributed MVC algorithm exists, our theorem rules out the existence of any corresponding local CapMDS approximation algorithm.

Locality-preserving reduction: Given a graph $G=(V,E)$ with $|V|=n$ and maximal degree Δ. Construct a graph $H_G=(V_H,E_H)$ as follows. Define two parameters

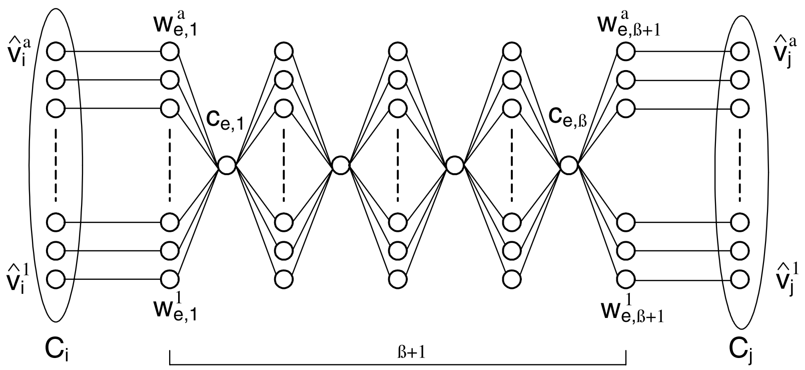

Fig. 3 Each edge $e = (v_i, v_j) \in E$ is replaced by an edge-chain of width $\beta + 1$ and height α. Nodes in clusters C_i and C_j form cliques

$\beta = 1/(2\epsilon)$ and $\alpha = \Delta/\epsilon$. For each node $v_i \in V$, H_G contains a clique C_i of α nodes $C_i = \{\hat{v}_i^1, \ldots, \hat{v}_i^\alpha\}$. For each edge $e_{ij} = (v_i, v_j) \in E$ in G, connect the cliques C_i and C_j as shown in Fig. 3: Node $\hat{v}_i^\ell \in C_i$ is connected to a node $w_{e,1}^\ell$, and node $\hat{v}_j^\ell \in C_j$ is connected to $w_{e,\beta+1}^\ell$. For every edge $e \in E$, H_G additionally contains $\beta - 1$ clusters each containing α nodes $w_{e,g}^\ell$, for $g = 2, \ldots, \beta$ and $\ell = 1, \ldots, \alpha$, and β *center-nodes* $c_{e,1}, \ldots, c_{e,\beta}$. For all $\ell = 1, \ldots, \alpha$, H_G contains edges $(c_{e,g}, w_{e,g}^\ell)$ and $(c_{e,g}, w_{e,g+1}^\ell)$. We call this construction for each edge in G an *edge-chain* in H_G. All nodes in H_G have capacity $\alpha + 1$.

For the locality preserving reduction from CapMDS to MVC, we need the following definitions. Given a graph G, let $\mathcal{O}_{VC}(G)$ and $\mathcal{O}_{Cap}(H_G)$ be the optimal solutions to the MVC problem on G and the CapMDS problem on corresponding H_G, respectively. Furthermore, let $\mathcal{A}_{VC}(G)$ be the MVC solution in G computed by a (possibly randomized) distributed algorithm with time complexity at most k, and let $\mathcal{A}_{Cap}(H_G)$ denote the solution to the CapMDS problem with $(1+\epsilon)$-capacity violations on H_G by a (possibly randomized) distributed algorithm with time complexity k/ϵ.

The first lemma bounds the relative size of the optimum solution in G and H_G.

Lemma 4.2 *For a graph G and its corresponding H_G, it holds that $\mathcal{O}_{Cap}(H_G) \le \frac{5}{2}\alpha \cdot \mathcal{O}_{VC}(G)$.*

Proof Given an optimal solution *OPT* to MVC, construct a solution S to CapMDS of size at most $3\alpha \cdot \mathcal{O}_{VC}(G)$ as follows. For every node $v_i \in V$ with value x_i in *OPT*, select αx_i nodes from the corresponding clique C_i in H_G and use these nodes to cover all of its adjacent nodes on each incident edge-chain. This is possible due to $\alpha \ge \Delta$. In addition, select all center-nodes $c_{e,g}$ for each edge $e \in E$ and finally, select one arbitrary node from each clique C_i, $v_i \in V$ to the capacitated dominating set.

The union of nodes thus selected can cover all nodes in H_G and hence, forms a feasible solution to the CapMDS problem on H_G. Specifically, because *OPT* is a feasible MVC solution, every edge in G is covered and hence, at least α nodes on each edge-chain in H_G are covered by nodes $\hat{v}_i^\ell \in C_i$. The remaining $\beta\alpha$ nodes in every edge-chain can be covered by its β selected center-nodes. Finally, a single node selected in each clique C_i, $v_i \in V$ suffices to cover the remaining clique nodes.

The number of edges in G (and hence, edge-chains in H_G) is at most $\Delta\mathcal{O}_{VC}(G)$. Because $\alpha x_i + 1$ nodes are selected from each clique C_i, the size of S is upper bounded by

$$\begin{aligned}|S| &\le \alpha\mathcal{O}_{VC}(G) + \beta\Delta\mathcal{O}_{VC}(G) + n\\ &\le \frac{5}{2}\alpha \cdot \mathcal{O}_{VC}(G),\end{aligned}$$

because of $n \le \Delta\mathcal{O}_{VC}(G)$ and the definitions of α and β. □

In the sequel, let $\rho_G(k)$ be defined as the maximum value such that for any randomized distributed algorithm *ALG* with expected running time at most k, it holds that $\mathcal{A}_{VC}(G) \ge \rho_G(k) \cdot \mathcal{O}_{VC}(G)$. In other words, $\rho_G(k)$ denotes the best achievable *approximation ratio* by any such k-local MVC algorithm on G. The following lemma shows that given an efficient (k/ϵ)-local CapMDS algorithm, a k-local MVC algorithm with good approximation ratio can be constructed.

Lemma 4.3 *Given a graph G and corresponding H_G, every (possibly randomized) distributed algorithm with time complexity at most* $\mathrm{O}(k/\epsilon)$ *produces a solution to the* $(1+\epsilon)$*-violated CapMDS problem of size* $\mathcal{A}_{Cap}(H_G) \ge \frac{1}{2}(\alpha-1)\rho_G(k) \cdot \mathcal{O}_{VC}(G)$.

Proof Assume for contradiction that there is a distributed algorithm ALG_{Cap} whose solution $\mathcal{A}_{Cap}(H_G)$ after k/ϵ communication rounds is of size less than $\frac{1}{2}(\alpha-1)\rho_G(k) \cdot \mathcal{O}_{VC}(G)$. Using ALG_{Cap} we can construct a distributed MVC algorithm ALG_{VC} for G with time complexity k whose solution is of size $\mathcal{A}_{VC}(G) < \rho_G(k)\mathcal{O}_{VC}(G)$, which contradicts the definition of $\rho_G(k)$.

Given ALG_{Cap}, ALG_{VC} proceeds as follows. Each node $v_i \in V$ collects in time k its entire k-hop neighborhood. It then locally constructs the (k/ϵ)-hop neighborhood of a graph H_G using an arbitrary assignment of identifiers to nodes $w^i_{e,g}$ and $c_{e,g}$ on its incident edge-chains and simulates ALG_{Cap} on this local part of H_G. Let $\mathcal{A}_{Cap}(H_G)$ be the outcome of this simulation. Without loss of generality, we can assume that only clique nodes $\hat{v}^\ell_i$ and center nodes $c_{e,g}$ are chosen as dominators in $\mathcal{A}_{Cap}(H_G)$. The reason is that any solution containing nodes $w^i_{e,g}$ can be transformed into a solution of less or equal size in which the dominating set is formed only of nodes $\hat{v}^\ell_i$ and $c_{e,g}$ in time $\mathrm{O}(\beta) = \mathrm{O}(1/\epsilon)$. Each node v_i then transforms the resulting CapMDS solution to a MVC solution by setting its local variable $a_i := \frac{2}{\alpha-1}\sum_{j=1}^{\alpha} x^\ell_i$, where x^ℓ_i is the primal value assigned to $\hat{v}^\ell_i$.

We now show that the assignment vector $\mathbf{a} = (a_1, \ldots, a_n)$ forms a feasible solution to the *MVC* problem on G. The number of nodes on each edge-chain e_{ij} is $(\beta+1)\alpha + \beta$. Without violating their capacity constraints, the center nodes $c_{e,g}$ can cover $\beta(\alpha+1)$ of these nodes in total. If each center node is allowed to violate its capacity constraints by a factor of $(1+\epsilon)$, the number of nodes on the edge-chain covered by center nodes is at most $\beta(\alpha+1)(1+\epsilon)$. Hence, at least $(\beta+1)\alpha + \beta - \beta(\alpha+1)(1+\epsilon) = \alpha - (\alpha+1)\beta\epsilon = \frac{\alpha-1}{2}$ nodes must be covered using clique nodes $x_i^\ell \in C_i$ and $x_j^\ell \in C_j$. For each edge $e_{ij} \in E$, it therefore holds that $\sum_{\hat{v}_i^\ell \in C_i} x_i^\ell + \sum_{\hat{v}_j^\ell \in C_j} x_j^\ell \geq \frac{\alpha-1}{2}$ and consequently, $a_i + a_j \geq 1$.

As $\mathcal{A}_{Cap}(H_G)$ consists of clique nodes and center nodes, it holds that $\mathcal{A}_{Cap}(H_G) > \sum_{v_i \in V} \sum_{\hat{v}_j^\ell \in C_i} x_j^\ell$. The size $\mathcal{A}_{VC}(G)$ of the resulting MVC solution is at most

$$\mathcal{A}_{VC}(G) = \sum_{v_i \in V} a_i \leq \sum_{v_i \in V} \left(\frac{2}{\alpha - 1} \sum_{\ell=1}^{\alpha} x_i^\ell \right)$$
$$< \frac{2}{\alpha - 1} \cdot \mathcal{A}_{Cap}(H_G).$$

The proof is now concluded by observing that if $\mathcal{A}_{Cap}(H_G) < \frac{\alpha-1}{2} \rho_G(k) \mathcal{O}_{VC}(G)$, then $\mathcal{A}_{VC}(G) < \rho_G(k) \mathcal{O}_{VC}(G)$, which contradicts the definition of $\rho_G(k)$. $\square$

Combining the results obtained in Lemmas 4.2 and 4.3 allows us to derive a hardness of distributed approximation result on CapMDS with $(1+\epsilon)$-violations by reduction to a known result on MVC.

Theorem 4.4 *For every $\epsilon \geq 1$ and some constant c, there are graphs H with n nodes such that every (possibly randomized) distributed algorithm with time complexity at most k/ϵ has an approximation ratio γ at least*

$$\gamma \in \Omega\left(\frac{n^{c/k^2}}{k} \right)$$

for the CapMDS problem in which capacities can be violated by a factor of $1+\epsilon$.

Proof It has been shown in [18] that there exist graphs G with n_G nodes and maximum degree Δ_G in which no (possibly randomized) distributed algorithm with running time k can achieve an approximation ratio better than $\rho_G(k) \in \Omega(n_G^{c'/k^2}/k)$. For such a graph G, construct the corresponding graph H_G. By Lemma 4.2, it holds that $\mathcal{O}_{Cap}(H_G) \leq \frac{5}{2}\alpha \cdot \mathcal{O}_{VC}(G)$. Furthermore, we know from Lemma 4.3 that $\mathcal{A}_{Cap}(H_G) \geq \frac{1}{2}\alpha \rho_G(k) \cdot \mathcal{O}_{VC}(G)$ for any distributed algorithm with running time at most k/ϵ. Finally, the number of nodes n in H_G is

$$n = \mathrm{O}\left(\frac{n_G \Delta \alpha}{\epsilon} \right) = \mathrm{O}\left(\frac{n_G \Delta^2}{\epsilon^2} \right) = \mathrm{O}\left(\frac{n_G^3}{\epsilon^2} \right).$$

Plugging in these values in the above bounds and adjusting c' to an appropriate c'' yields a lower bound of $\Omega((\epsilon n)^{c''/k^2}/k)$. Because $n > n_G/\epsilon^2$, we know that $\epsilon > 1/\sqrt{n}$ and we obtain the theorem by appropriately adapting the constant c''. $\square$

Finally, solving the approximability lower bound of Theorem 4.4, we can derive the following time lower bound on any distributed algorithm that achieves a polylogarithmic approximation to the CapMDS problem with $(1+\epsilon)$-violations.

Corollary 4.5 *Consider an $\epsilon \geq 1$. Every distributed algorithm for the CapMDS problem that achieves an approximation ratio of $\mathrm{O}(polylog(n))$ and violates capacity constraints by at most a factor of $(1+\epsilon)$ must have a time complexity of at least $\Omega(\frac{1}{\epsilon}\sqrt{\frac{\log n}{\log\log n}})$ communication rounds.*

In the following subsection, we present a distributed algorithm whose approximation ratio is within a polylogarithmic ratio of this lower bound.

4.2 Upper Bounds

The lower bound established by Theorem 4.4 is much weaker than the one in Theorem 4.1 which raises hope that, indeed, moderate relaxations of the capacity constraints can significantly improve the approximability of the problem. In this section, we present a local distributed approximation algorithm that runs in $\mathrm{O}(\log^3(n) + \log(n)/\epsilon)$ communication rounds and computes an $\mathrm{O}(\log\Delta)$ approximation, while violating the capacity constraints by a factor of at most $1+\epsilon$.

The algorithm works in two phases. First, we compute a solution for the linear programming relaxation of the integer LP $\mathrm{ILP}_{\mathrm{CapDS}}$. This results in a fractional solution to the CapMDS problem where the variables x_i and y_{ij} can have arbitrary values in $[0,1]$, i.e., (4) of $\mathrm{ILP}_{\mathrm{CapDS}}$ is relaxed to $x_i, y_{ij} \in [0,1]$. We denote the LP version of $\mathrm{ILP}_{\mathrm{CapDS}}$ by $\mathrm{LP}_{\mathrm{CapDS}}$. In a second phase, we then round this fractional solution to an integer one. Both phases make use of a distributed network decomposition algorithm which was proposed by Linial and Saks in [26]. The rest of this section is organized as follows. We first summarize the main properties of the network decomposition described in [26]. Using this algorithm, we then show how to compute an approximate solution for $\mathrm{LP}_{\mathrm{CapDS}}$ and we show how to round the fractional solution to an integer one.

Distributed Network Decomposition

Let $G=(V,E)$ be a graph with $n=|V|$ nodes. The basic building block of the algorithm in [26] is a randomized algorithm $\mathcal{A}(p,R)$ which computes a subset $S\subseteq V$ as well as a mapping of each node $u\in S$ to some leader node $\ell(u)\in V$. The following properties hold for arbitrary parameters $p\in[0,1]$ and $R\geq 1$:

1. $\forall u\in S: d_G(u,\ell(u))\leq R$.
2. $\forall u,v\in S: \ell(u)\neq\ell(v) \implies (u,v)\notin E$.
3. S can be computed in $\mathrm{O}(R)$ rounds.
4. $\forall u\in V: \mathbb{P}[u\in S]\geq p(1-p^R)^n$.

Thereby $d_G(u,v)$ denotes the distance between two nodes u and v in G. Algorithm $\mathcal{A}(p,R)$ computes a set of clusters of nodes such that nodes of different clusters are at distance at least 2 and any two nodes of the same cluster are at distance at most $2R$ in G. Moreover, every node belongs to some cluster with probability at least $p(1-p^R)^n$. Note that the algorithm does not bound the diameter of the graph which is induced by the nodes of a given cluster. It merely bounds the distance on G between any two nodes of the same cluster. The maximal distance d_G in a graph G between any two nodes of a cluster of nodes of G is called the *weak diameter* of the cluster.

We will invoke $\mathcal{A}(p, R)$ on graphs G^k where G is the graph on which we want to solve CapMDS. Because nodes in G^k are connected by an edge whenever their distance in G is at most k, a single communication round on G^k can be simulated by k communication rounds on G. When applied to G^k, Algorithm $\mathcal{A}(p, R)$ therefore computes clusters of weak diameter kR and the distance between two nodes in different clusters is larger than k (both with regard to G). The time complexity of $\mathcal{A}(p, R)$ on G^k is $\mathrm{O}(kR)$.

Solving the Linear Program

The basic approach for our distributed solution of the LP relaxation of CapMDS is as follows. We use the clusters constructed by Algorithm $\mathcal{A}(p, R)$ to divide the given LP into a set of smaller LPs which can be solved efficiently by a distributed algorithm. By choosing the parameters p and R in the right way and by doing sufficiently many parallel executions of $\mathcal{A}(p, R)$, we can combine the solutions of all small local LPs to obtain an approximate solution of the original global LP. This technique has already been applied successfully for the LP of the classical dominating set problem without capacities in [20].

Let $S \subseteq V$ be the subset of nodes which is selected in an execution of $\mathcal{A}(p, R)$ on G^2. Consider a subproblem $\mathrm{LP}_{\mathrm{CapDS}}[S]$ of the original LP $\mathrm{LP}_{\mathrm{CapDS}}$ where only nodes in S have to covered, that is, we change $\mathrm{LP}_{\mathrm{CapDS}}$ such that Condition (1) ($\sum_{v_j \in \Gamma(v_i)} y_{ji} \geq 1$) only needs to hold for all nodes $v_j \in S$. Lemma 4.6 shows that $\mathrm{LP}_{\mathrm{CapDS}}[S]$ can be solved efficiently.

Lemma 4.6 *Given the node set S of an execution of $\mathcal{A}(p, R)$, an optimal solution of* $\mathrm{LP}_{\mathrm{CapDS}}[S]$ *can be computed in* $4R + 2$ *rounds.*

Proof Let C_i for $i = 1, \dots, C(S)$ be the clusters induced by S where $C(S)$ denotes the number of clusters of S. Because nodes of different clusters C_i and C_j are at distance at least 3, the sets of nodes which can cover nodes in C_i and the set of nodes which can cover nodes in C_j are disjoint. We can therefore solve $\mathrm{LP}_{\mathrm{CapDS}}[S]$ by solving $\mathrm{LP}_{\mathrm{CapDS}}[C_i]$ for each cluster C_i and sum the solutions of all clusters. Let ℓ_i be the leader node of a cluster C_i, i.e., $u \in C_i \to \ell(u) = \ell_i$. Because the distance on G^2 between any node $u \in C_i$ and ℓ_i is at most R, all information needed to solve $\mathrm{LP}_{\mathrm{CapDS}}[C_i]$ is at distance $2R + 1$ from ℓ_i. In $4R + 2$ communication rounds, node ℓ_i can therefore collect its complete $(2R + 1)$-neighborhood, locally compute an optimal solution of $\mathrm{LP}_{\mathrm{CapDS}}[C_i]$, and send this information back to the nodes of its $(2R + 1)$-neighborhood. Given the set S, we can therefore compute an optimal solution for $\mathrm{LP}_{\mathrm{CapDS}}[S]$ in time $4R + 2$. □

The following theorem shows that by invoking $\mathcal{A}(p, R)$ sufficiently many times with the right choices of p and R, $\mathrm{LP}_{\mathrm{CapDS}}$ can be well approximated if we allow a small violation of the capacities.

Theorem 4.7 *If capacities are allowed to be violated by a factor of* $1+\epsilon$, *it is possible to compute a* $(1+\epsilon)$*-approximation of the linear programming relaxation of CapMDS in* $\mathrm{O}(\log(n)/\epsilon)$ *rounds with high probability.*

Proof Consider K independent, parallel executions of $\mathcal{A}(p,R)$ on G^2 that result in K node sets $S_1,\dots,S_K$. Let k_i be the number of times, a node v_i occurs in one of the sets, i.e., $k_i=|\{h\in[K]|v_i\in S_h\}|$. Further let $k_{\min}=\min_i k_i$. Assume that we choose K large enough such that $k_{\min}\geq 1$. By Lemma 4.6, we can solve all LPs $\mathrm{LP}_{\mathrm{CapDS}}[S_h]$ for $h\in[K]$ in $4R+2$ rounds. Note that we can solve all K LPs in parallel. Let $x_i^{(h)}$ and $y_{ij}^{(h)}$ be the values of the variables x_i and y_{ij} in the solution of $\mathrm{LP}_{\mathrm{CapDS}}[S_h]$. As a first step we combine the solutions of $\mathrm{LP}_{\mathrm{CapDS}}[S_h]$ to get a solutions of $\mathrm{LP}_{\mathrm{CapDS}}$ as follows. We sum the variables of the LPs $\mathrm{LP}_{\mathrm{CapDS}}[S_h]$ to get variables $\hat{x}_i$ and $\hat{y}_{ij}$:

$$\forall v_i,v_j\in V\colon\quad \hat{x}_i=\sum_{h=1}^{K}x_i^{(h)}\quad\text{and}\quad \hat{y}_{ij}=\sum_{h=1}^{K}y_{ij}^{(h)}.$$

Because Inequalities (2) and (3) hold for the variables $x_i^{(h)}$ and $y_{ij}^{(h)}$, they also hold for the variables $\hat{x}_i$ and $\hat{y}_{ij}$. Inequality (1) which states that every node has to be covered becomes

$$\forall v_i\in V\colon\quad \sum_{v_j\in\Gamma(v_i)}\hat{y}_{ji}\geq k_i. \tag{5}$$

In order to have every node be covered exactly once, we define

$$y_{ij}:=\frac{\hat{y}_{ij}}{k_j}\quad\text{and}\quad x_i:=\hat{x}_i\cdot\max_{v_j\in\Gamma(v_i)}\frac{y_{ij}}{\hat{y}_{ij}}\leq\frac{\hat{x}_i}{k_{\min}}. \tag{6}$$

Inequalities (1) and (3) are now satisfied and instead of Inequality (2), the following inequality holds for all $v_i\in V$:

$$\begin{aligned}\sum_{v_j\in\Gamma(v_i)}y_{ij}&\leq\frac{\max_{v_j\in\Gamma(v_i)}k_j}{\min_{v_j\in\Gamma(v_i)}k_j}\cdot x_i\cdot\mathrm{cap}(v_i)\\&\leq\frac{K}{k_{\min}}\cdot x_i\cdot\mathrm{cap}(v_i).\end{aligned}$$

We therefore have to allow to violate the capacity constraints by a factor of $K/k_{\min}$. For $v_i\in V$, let x_i^* be the value of variable x_i in an optimal solution of $\mathrm{LP}_{\mathrm{CapDS}}$. Recall that $x_i^{(h)}$ is the value of x_i in an optimal solution of $\mathrm{LP}_{\mathrm{CapDS}}[S_h]$. For all h, we therefore get $\sum_i x_i^{(h)}\leq\sum_i x_i^*$ and thus by Inequality (6),

$$\sum_{v_i\in V}x_i\leq\frac{1}{k_{\min}}\cdot\sum_{h=1}^{K}\sum_{v_i\in V}x_i^{(h)}\leq\frac{K}{k_{\min}}\cdot\sum_{v_i\in V}x_i^*.$$

By allowing to violate the capacities by a factor of $K/k_{\min}$, we can therefore compute a $K/k_{\min}$-approximation in time $\mathrm{O}(R)$. It remains to set p, R, and K such that $K/k_{\min} \le 1+\epsilon$ w.h.p.

Let $p = e^{-\alpha\epsilon}$, $R = \beta \ln(n)/\epsilon$, and $K = \gamma \ln(n)/\epsilon^2$ for constants α, β, and γ, and let $q := p(1-p^R)^n$. For any node v_i and any set S_h, we have $\mathbb{P}[v_i \in S_h] \ge q$. By choosing p and R as given, we obtain

$$q = e^{-\alpha\epsilon} \cdot \left(1 - \frac{1}{n^{\alpha\beta}}\right)^n \ge e^{-2\alpha\epsilon} \ge 1 - 2\alpha\epsilon \tag{7}$$

if we choose β such that $n^{\alpha\beta} \ge (1+\mathrm{o}(1)) \cdot n/(\alpha\epsilon)$. Note that we can assume that $\epsilon \le n$ (the lemma becomes trivial for $\epsilon > n$) and therefore get $\beta \in \mathrm{O}(1)$ if $\alpha \in \mathrm{O}(1)$. Let $\mu(k_i) \ge q \cdot K$ be the expected value of k_i. We can bound the probability that $k_i < (1-\delta)\mu(k_i)$ by using the Chernoff inequality and Inequality (7):

$$\mathbb{P}[k_i < (1-\delta)\mu(k_i)] \le e^{-\frac{\mu(k_i)\delta^2}{2}} \le e^{-\frac{(1-2\alpha\epsilon)K\delta^2}{2}}.$$

We choose $K = \gamma \ln(n)/\epsilon^2$ and $\delta \in \mathrm{O}(\epsilon)$ such that $1/(1+\epsilon) = (1-\delta)(1-2\alpha\epsilon)$ and obtain

$$\mathbb{P}\left[\frac{K}{k_{\min}} > 1+\epsilon\right] \le n \cdot e^{-\frac{(1-2\alpha\epsilon)K\delta^2}{2}} \le \frac{1}{n^c}.$$

for an arbitrary constant c by using a union bound argument and by choosing the constants α and γ appropriately. □

Rounding

Having found a solution for $\mathrm{LP}_{\mathrm{CapDS}}$, the next step is to convert the computed fractional solution into an integer one. For this, we develop a novel distributed randomized rounding technique that consists of two steps. First, we only round the variables x_i to values in $\{0,1\}$ but still allow the variables y_{ij} to be fractional. Thus, we select a dominating set but still allow the assignment to be fractional. We then round the assignment of nodes to dominators in a second step.

While solving the LP only requires a small multiplicative violation of the capacity constraints, we also have to allow additive violations of the capacity constraints for the rounding. We say that the capacity constraints of a solution of $\mathrm{LP}_{\mathrm{CapDS}}$ are (ρ, β)-violated if instead of Inequality (2), we have

$$\forall v_i \in V\colon \quad \sum_{v_j \in \Gamma(v_i)} y_{ij} \le x_i \cdot \big(\rho \cdot \mathrm{cap}(v_i) + \beta\big). \tag{8}$$

Assume that we start the rounding process with an α-approximate solution of $\mathrm{LP}_{\mathrm{CapDS}}$ with (ρ, β)-violated capacity constraints. Let us denote the variables before the rounding by x_i and y_{ij} and the variables after the rounding by x_i' and y_{ij}'. Recall that before the rounding each node v_i only knows its own (fractional) variables x_i and y_{ij}, for all neighbors v_j.

To round the variables x_i, we use a standard randomized rounding technique which has been introduced in [30] and which has been adapted to a distributed context in [22]. The basic idea is to interpret the values of the variables x_i as probabilities for the nodes to join the dominating set. The details of this distributed randomized rounding are given by Algorithm 1 which is executed by all nodes v_i.

Lemma 4.8 *When applied to an α-approximate solution with (ρ, β)-violated capacities, Algorithm 1 computes a new solution of* $\mathrm{LP}_{\mathrm{CapDS}}$ *with integer variables x_i' and $(\rho, \beta+1)$-violated capacity constraints in* 1 *round. The expected approximation ratio of the computed solution is* $\alpha \ln(\Delta+1)+1$.

Proof Let us first consider the situation after Line 4 of Algorithm 1. The violation of the capacity constraints is not affected by changing the variables x_i and y_{ij} to x_i' and y_{ij}' in Lines 2 and 3, respectively. Because x_i and y_{ij} are multiplied by the same factor, Inequality (8) holds for the variables x_i' and y_{ij}' if it holds for the variables x_i and y_{ij}. The expected number of nodes in the dominating set after Line 4 is $\ln(\Delta+1) \cdot \sum_i x_i$.

Let q_i be the probability that a node v_i is not covered after Line 4, i.e., q_i is the probability that the condition in the if statement in Line 5 is true for v_i. The probability q_i can be bounded as follows (δ_i denotes the degree of v_i):

$$\begin{aligned} q_i &= \prod_{v_j \in \Gamma(v_i)} \left(1 - p_j \frac{y_{ij}}{x_i}\right) \\ &\le \prod_{v_j \in \Gamma(v_i)} \big(1 - y_{ij} \ln(\Delta+1)\big) \\ &\le \left(1 - \frac{\sum_{v_j \in \Gamma(v_i)} y_{ij} \ln(\Delta+1)}{\delta_i + 1}\right)^{\delta_i+1} \\ &\le \left(1 - \frac{\ln(\Delta+1)}{\delta_i+1}\right)^{\delta_i+1} \le e^{-\ln(\Delta+1)} \\ &= \frac{1}{\Delta+1}. \end{aligned}$$

Algorithm 1 Selecting the Dominating Set

```
1: $p_i := \min\{1, x_i \cdot \ln(\Delta+1)\}$
2: $x_i' := \begin{cases} 1 & \text{with probability } p_i \\ 0 & \text{otherwise} \end{cases}$
3: $\forall v_j \in \Gamma(v_i) : y_{ij}' := y_{ij} \cdot x_i'/x_i$
4: send $x_i', y_{ij}'$ for $v_j \in \Gamma(v_i)$ to all neighbors
5: if $\sum_{v_j \in \Gamma(v_i)} y_{ji} < 1$ then
6:     $x_i' := 1$; $y_{ii}' := 1$
7: fi
```

The expected number of nodes which are added to the dominating set in Line 6 therefore is at most $n/(\Delta+1)$. Note that since every node can cover at most $\Delta+1$ nodes, every capacitated dominating set has size at least $n/(\Delta+1)$. The expected approximation ratio of the computed solution therefore is $\alpha\ln(\Delta+1)+1$ as claimed.

To conclude the proof, let us now consider the capacity violation caused by Line 6. Assume that a node v_i has to cover itself in Line 6. If v_i has already set $x_i'=1$ but has not covered itself before coming to Line 6, it might already exhaust its capacity. Setting $y_{ii}'=1$ in Line 6 can therefore cause an additional additive capacity violation of 1. □

Having selected the nodes for the capacitated dominating set, it remains to convert the fractional assignment of nodes to dominators into an integer assignment. We first consider the problem of rounding the assignments in a non-distributed fashion and then show how to obtain a distributed algorithm.

W.l.o.g., we can assume that nodes are covered exactly once, i.e., we can assume that Inequality (1) holds with equality, because, if for some v_i, $\sum_{v_j\in\Gamma(v_i)} y_{ji}'>1$, we can decrease the values of the variables y_{ji}' in order to obtain equality. Let $D=\{v_i\,|\,x_i'=1\}$ be the nodes selected as dominators. Consider the following directed acyclic graph H. Graph H has $|D|+n+2$ nodes, a node p_i for every $v_i\in D$, a node q_i for every $v_i\in V$, and two nodes s and t. There is an arc from s to every node p_i, there is an arc from a node p_i to q_j if there is an edge between v_i and v_j in G, and there is an arc between every node q_i and t.

The problem of assigning nodes to dominators can be considered as a maximal (s,t)-flow problem on H in which the flow capacities of arcs from s to p_i are $\mathrm{cap}(v_i)$ (or $\rho\mathrm{cap}(v_i)+\beta$ if we allow a (ρ,β)-violation of the capacity constraints) and the flow capacities of all other arcs are 1. The fractional assignments y_{ij}' induce the following flow on H. The flow on an arc from p_i to q_j is y_{ij}' and consequently, the flow from s to a node p_i is $\sum_{v_j\in\Gamma(v_i)} y_{ij}'$ and the flow from nodes q_i to t is 1.

The fractional flow induced by the fractional assignment given by the variables y_{ij}' can be converted into an integer flow step-by-step using the following simple algorithm. Let us call an arc with a fractional flow, a fractional arc. Note that the total flow at each node of H is an integer (the total flow is n at s and t and 0 at all inner nodes). Therefore, every fractional arc of H must lie on a cycle consisting only of fractional arcs. Consider such a cycle $\mathcal{C}=(a_1,\dots,a_t)$ of fractional arcs. Let a_i be the arc of $\mathcal{C}$ whose flow is closest to an integer value and let ξ_i be the difference between the flow of a_i and this integer value. We can traverse the cycle $\mathcal{C}$ and depending on the direction in which we traverse an arc a_j add $+\xi_i$ or $-\xi_i$ to the flow of a_j such that the total flow at every node remains the same and such that the flow of a_i is rounded to an integer. Note that by the choice of arc a_i and because the flow capacities of all arcs are integers, we do not violate any of the capacities when adapting the flow values like that. The described method eventually constructs an integer flow because in each step, the number of fractional arcs of H is reduced by at least one.

The above rounding scheme is inherently centralized and when trying to implement it using a *distributed algorithm*, we face two main problems. First, in k rounds, nodes can only detect cycles of length at most $2k$. Second, even if all fractional cycles were short, in order to have enough parallelism, a large number of cycles needs to be handled simultaneously. Unfortunately, the only solution for the first problem is to only look at short cycles and to round fractional arcs which do not lie on short cycles by some other method. The second problem can be solved by again using the network decomposition algorithm from [26] described at the beginning of this section. The details of the distributed rounding mechanism for node assignments are defined by Algorithm 2. The algorithm executes the decomposition algorithm $\mathcal{A}(p, R)$ K times on a graph $G^{\mathrm{O}(\log n)}$. We use the following notation. Denote the set of nodes which is selected in execution h by S_h and let $C_i^{(h)}$, $i = 1, \dots, C(S_h)$ be the clusters induced by S_h. Further let $\ell_i^{(h)}$ be the leader of cluster $C_i^{(h)}$. For a cluster C, define $H[C]$ to be the subgraph of H induced by s, t, and all nodes p_i and q_i corresponding to nodes $v_i \in C$. Finally, Algorithm 2 uses two constants c and d which we will specify later.

Lemma 4.9 *When applied to an α-approximation of* $\mathrm{LP}_{\mathrm{CapDS}}$ *with integer variables x_i and (ρ, β)-violated capacities, Algorithm 2 computes a 3α-approximation of* $\mathrm{ILP}_{\mathrm{CapDS}}$ *with capacity violation $(\rho, \beta + 1)$ in time* $\mathrm{O}(\log^3 n)$ *w.h.p.*

Proof When choosing $p = 1/2$ and $R = \log_2(n + 1)$, we have $p(1 - p^R)^n \geq 1/(2e)$. The number K of executions of $\mathcal{A}(p, R)$ is chosen such that every node $v \in V$ is in a

Algorithm 2 Assigning Nodes to Dominators

```
1:  $p := 1/2$; $R := \log_2(n+1)$; $K := 2e \cdot c \cdot \ln n$
2:  $K$ executions of $\mathcal{A}(p, R)$ on $G^{d \ln n}$
3:  for $h := 1$ to $K$ do
4:      for all clusters $C_i^{(h)}$ do
5:          for all nodes $u \in C_i^{(h)}$ do
6:              for all nodes $v : d_G(u, v) \leq \frac{d \ln n}{2}$ do
7:                  Add $v$ to $C_i^{(h)}$
8:              od
9:          od
10:         $\ell_i^{(h)}$ collects induced subgraph of $C_i^{(h)}$
11:         remove all fractional cycles in $H[C_i^{(h)}]$
12:     od
13: od
14: for all $(v_i, v_j) \in E$ do
15:     if $y'_{ij} \notin \{0, 1\}$ then
16:         $y'_{ij} := 0$
17:     fi
18: od
19: for all $v_i \in V$ do
20:     if $\sum_{v_j \in \Gamma(v_i)} y'_{ji} < 1$ then
21:         $x'_i := 1$; $y'_{ii} := 1$
22:     fi
23: od
```

cluster of $G^{d\ln n}$ at least once w.h.p. when choosing $c \geq 1$:

$$\begin{aligned}&\mathbb{P}\left[\exists v \in V : \forall h \in [K] : v \notin S_h\right]\\&\leq n \cdot \left(1 - p(1-p^R)^n\right)^K\\&\leq \left(1 - \frac{1}{2e}\right)^{2e\cdot c\cdot \ln n} < \frac{1}{n^c}.\end{aligned}$$

Let us therefore assume that every node $v \in V$ is in at least one cluster of $G^{d\ln n}$. In Lines 5–9, for each node v in a cluster $C_i^{(h)}$, all nodes at distance $\leq d\ln(n)/2$ from v (the $(d\ln(n)/2)$-ball of v) is added to cluster $C_i^{(h)}$. Because the distance between nodes of different clusters of S_h are at distance more than $d\ln n$ in G, two clusters $C_i^{(h)}$ and $C_j^{(h)}$ remain disjoint even after adding all $(d\ln(n)/2)$-balls. This means that every $(d\ln(n)/2)$-ball of G is completely contained in some cluster $C_i^{(h)}$.

In Line 11, the rounding algorithm which we described for the non-distributed case is applied to subgraphs $H[C_i^{(h)}]$ of H induced by clusters $C_i^{(h)}$. Because every ball of radius $d\ln(n)/2$ of G is contained in some cluster, every cycle of length $d\ln(n)$ of G is also completely contained in some cluster. Therefore, also every cycle of length $d\ln(n)$ of H is completely contained in some $H[C_i^{(h)}]$. After applying the rounding step in Line 11 for all clusters (for all i and h), H does not contain fractional cycles of length at most $d\ln(n)$ any more.

It is well-known that the number of edges of every graph G with n nodes and girth g (smallest cycle has length g) is upper bounded by $n^{1+\tau/g}$ for some constant τ. Let V_f be the set of nodes which are still covered by fractional edges after Line 13 and let D be the initial dominating set. The number of remaining fractional edges $e_f(H)$ of H after Line 13 is at most

$$e_f(H) \leq (1 + |D| + |V_f|)^{1+\frac{\tau}{d\ln n}}. \tag{9}$$

Because every node in V_f is covered by at least 2 fractional edges, we also have

$$e_f(H) \geq 2 \cdot |V_f|. \tag{10}$$

Combining (9) and (10) gives $|V_f| \leq 2|D|$ if we choose $d = \tau(\ln 3 + o(1))$. Note that the nodes which are added to the dominating set in Line 21 are exactly the nodes in V_f. The size of the dominating set is therefore increased by at most a factor of 3. The only possible additional capacity constraint violation of Algorithm 2 is caused when a node v_i has to cover itself in Line 21.

We have therefore proven the claimed approximation ratio and capacity constraint violation and it remains to show that the time complexity of Algorithm 2 is $\mathrm{O}(\log^3 n)$. The number of rounds needed to compute the K executions of $\mathrm{A}(p, R)$ is $\mathrm{O}(Rd\ln n) = \mathrm{O}(\log^2 n)$ when done in parallel and $\mathrm{O}(KRd\ln n) = \mathrm{O}(\log^3 n)$ when done sequentially. Lines 3–13 have to be done sequentially for $h = 1, \ldots, K$. However for a particular h, all clusters can be handled concurrently. The time to add the $(d\ln(n)/2)$-balls to the clusters, to collect the topology of each cluster, and to do the local computation at the cluster leaders is $\mathrm{O}(Rd\ln n) = \mathrm{O}(\log^2 n)$. The total time complexity for Lines 3–13 therefore becomes $O(K \log^2 n) = O(\log^3 n)$. The rest of the algorithm can be computed in a constant number of rounds. □

Combining Theorem 4.7 and Lemmas 4.8 and 4.9, we obtain the main theorem of this section.

Theorem 4.10 *There is a distributed capacitated dominating set algorithm with expected approximation ratio* $O(\log \Delta)$ *and time complexity* $O(\log^3 n + \log n/\epsilon)$ *w.h.p. if we allow* $(1+\epsilon, 2)$*-violated capacity constraints.*

Message Size: Our distributed capacitated dominating set algorithm makes frequent use of the assumption that we can pack as much information as we like in every message. Many of the computations such as computing a network decomposition of a graph G^k, however, can be implemented with messages of size only $O(\log n)$ if we allow the time complexity to grow by a polylogarithmic factor [26]. However there are parts of the algorithm for which we do not know how to implement them with small messages in polylogarithmic time. In particular, this is true for solving the local LPs in the $\mathrm{LP}_{\mathrm{CapDS}}$ approximation algorithm and for removing short fractional cycles of H in the second rounding step. Whether a result similar to the one given by Theorem 4.10 can be achieved with messages of polylogarithmic size is an interesting open problem.

Alternative Rounding: For $\epsilon \gg 1/\log^2 n$, the main contribution to the time complexity of our CapMDS algorithm comes from the rounding of the assignments. If we allow to violate the capacity constraints by a multiplicative constant factor, it is possible to compute integer assignments in $O(1)$ rounds by using the following simple method. Each node v_j chooses a dominator v_i according to the distribution given by the values of y_{ij} and requests to be covered by v_i. Each dominator v_i accepts γ such requests for a suitable constant γ. If v_i does not accept the dominating request from a node v_j, v_j joins the capacitated dominating set and covers itself. It can be shown that when choosing γ large enough, the number of nodes added to the dominating set is proportional to the number of nodes already in the dominating set w.h.p.

5 Unit Disk and Bounded Independence Graphs

5.1 Lower Bounds

In the non-uniform case, the CapMDS problem remains non-local even in geometric graphs such as the unit disk graphs or general BIGs. The reason is that by appropriately adjusting capacities, the example in Fig. 2 can be drawn as a BIG.

Theorem 5.1 *There are bounded independence graphs G with non-uniform capacities, such that in k communication rounds, every (possibly randomized) distributed algorithm for the minimum capacitated dominating set problem on G has approximation ratios at least* $\Omega(n/k^2)$ *and* $\Omega(\Delta/k)$.

Proof Consider the example in Fig. 2, but collapse every cluster to a clique. Arrange these cliques in such a way that bridge-nodes cover all nodes in their neighboring cliques. Every node in a clique has capacity 1, whereas the capacity of bridge-nodes remains $m+1$. The remainder of the proof is then analogous to the proof of Theorem 4.1. □

5.2 Upper Bounds for the Uniform Case

We have seen that for general graphs, even for uniform capacities, it is impossible to compute a non-trivial approximation for CapMDS in time substantially less than the diameter of the network without violating the capacity constraints. We now show that the situation is different for geometric network graphs G by presenting an efficient distributed algorithm which computes a constant approximation for CapMDS with *uniform capacities* without violating capacities if G is a BIG.

In the following we assume that $\text{cap}(v) = \text{cap}$ for all $v \in V$. Our algorithm (Algorithm 3) for computing a small capacitated dominating set on BIGs consists of two phases. In the first phase, a maximal independent set (MIS) S of G is computed. S induces a partition of the nodes of G into clusters C_i of radius 1 as follows. We construct a cluster C_i for every node $v_i \in S$. Every node $v \in V \setminus S$ is added to an arbitrary cluster C_i of an adjacent node $v_i \in S$. In the second phase, for each cluster C_i, a capacitated dominating set of size $\mathrm{O}(|C_i|/\text{cap})$ is computed. The details of the computation are defined in Algorithm 3 which returns a set of pairs (u, C) consisting of a node u of the dominating set and the set of nodes C assigned to u. In Line 2 and 11, Algorithm 3 calls a function $\text{cluster}(S, V)$ which computes the clustering induced by a MIS S. We assume that the clusters (u_i, C_i) returned by $\text{cluster}(S, V)$ are sorted according to non-decreasing cluster size $|C_i|$.

Theorem 5.2 *Let G be a graph of bounded independence. Algorithm 3 computes an* $\mathrm{O}(1)$*-approximation of the minimum capacitated dominating set problem in time* $\mathrm{O}(T_{\text{MIS}})$*, where T_{MIS} denotes the time to compute a maximal independent set on G.*

Proof Assume that G is f-independence bounded for some function $f(r)$. We show that Algorithm 3 computes a capacitated dominating set D of size

$$|D| \le |S| + n \cdot f(1)/\text{cap}. \tag{11}$$

It is well known that a MIS S is an $f(1)$-approximation of the minimum dominating set problem without capacity constraints. For an optimal dominating set D^*, every node $u \in S$ can be assigned to an adjacent node $v \in D^*$ and by the definitions of a MIS and f-independence boundedness, at most $f(1)$ nodes $u \in S$ are assigned to every node $v \in D^*$. Because in a capacitated dominating set D, every node $v \in D$ can dominate at most $\text{cap}(v) = \text{cap}$ nodes, Inequality (11) implies that the set D is a $2f(1)$-approximation for CapMDS.

To prove Inequality (11), we show that the set C of every recursive call of $\text{CapDS}(u, C)$ in Lines 20 and 30 of Algorithm 3 contains at most

$$|C| \ge \frac{\text{cap}}{f(1)} \tag{12}$$

Algorithm 3 Dominating Set of Cluster C_i

```
1:  $S := \mathrm{MIS}(V)$
2:  $\{(u_i, C_i)\} := \mathrm{cluster}(S, V)$
3:  return $\bigcup_{u_i \in S} \mathrm{CapDS}(u_i, C_i)$
4:
5:  function $\mathrm{CapDS}(v, C_v)$
6:  begin
7:    if $|C_v| \leq \mathrm{cap}$ then
8:      return $\{(v, C_v)\}$
9:    else
10:     $S_v := \mathrm{MIS}(C_v \setminus \{v\})$
11:     $\{(u_i, C_i)\} := \mathrm{cluster}(S_v, C_v)$
12:     $t := |\{(u_i, C_i)\}|$
13:     if $t \leq 2$ then
14:       $A \subset C_t \setminus \{u_t\}$ s.t. $|C_v \setminus C_t \cup A| = \lceil \mathrm{cap}/2 \rceil$
15:       if $t = 1$ then
16:         $C_0 := A \cup \{v\}$
17:       else
18:         $C_0 := C_1 \cup A \cup \{v\}$
19:       fi
20:       return $\mathrm{CapDS}(v, C_0) \cup \mathrm{CapDS}(u_t, C_t)$
21:     fi
22:     $C_{t+1} := C_1 \cup \{v\}$; $u_{t+1} := v$; $i := 2$
23:     while $|C_{t+1}| + |C_i| \leq \mathrm{cap}$ do
24:       $C_{t+1} := C_{t+1} \cup C_i$; $i := i + 1$
25:     od
26:     if $|C_{t+1}| < \mathrm{cap}/t$ then
27:       $A \subset C_i \setminus \{u_i\}$ s.t. $|A| = \lceil \mathrm{cap}/t \rceil - |C_{t+1}|$
28:       $C_i := C_i \setminus A$; $C_{t+1} := C_{t+1} \cup A$
29:     fi
30:     return $\bigcup_{j=i}^{t+1} \mathrm{CapDS}(u_j, C_j)$
31:   fi
32: end
```

nodes. This means that every node $u \in D \setminus S$ covers at least $\mathrm{cap}/f(1)$ nodes, which implies Inequality (11).

Consider an execution of the function CapDS. Because G is f-independence bounded, we have $t \leq f(1)$. First consider the case $t \leq 2$. Because $C_v > \mathrm{cap}$ (otherwise there is no recursive call), sets C_0 and C_t in the recursive calls in Line 20 contain at least $\lceil \mathrm{cap}/2 \rceil$ elements. Note that we can assume that $f(1) \geq 2$ because otherwise G would be a complete graph. Inequality (12) is thus satisfied if $t \leq 2$.

If $t \geq 3$, Inequality (12) holds for $j = t + 1$ because set C_{t+1} is constructed such that $|C_{t+1}| \geq \lceil \mathrm{cap}/t \rceil$. For $i \leq j \leq t$ where i is the index for which the while loop condition in Line 23 is violated, we first consider the case where the condition in the if statements in Line 26 is not satisfied. Recall that we assume $|C_j| \leq |C_{j+1}|$ for all j. For the sake of contradiction, assume that $|C_j| < \mathrm{cap}/t$ for $j \geq i$. We then have

$$\sum_{h=1}^{j} |C_h| < j \cdot \frac{\mathrm{cap}}{t} < t \cdot \frac{\mathrm{cap}}{t} < \mathrm{cap}.$$

This is a contradiction to the assumption that the while loop condition in Line 23 is violated for $j = i$. We therefore know that Inequality (12) is satisfied for all j if the if-condition in Line 26 is not true. If the condition in Line 26 is satisfied, Inequality (12) holds for $j > i$ by using the same argument and it remains to prove that it also holds for $j = i$ in this case. By combining the while-condition of Line 23 and the if-condition of Line 26, we obtain $|C_i| > \mathrm{cap} - \mathrm{cap}/t - |A|$. Before adding $|A|$, the set C_{t+1} contains at least the two nodes v and u_1. We therefore have $|A| \leq \lceil \mathrm{cap}/t \rceil - 2 < \mathrm{cap}/t$ and thus $|C_i| > \mathrm{cap}/t$ if $t \geq 3$, which concludes the proof of Inequality (12), and hence the approximation ratio.

For the time complexity, note that all computations in function $\mathrm{CapDS}(v, C_v)$ are within v's 1-neighborhood. MIS nodes $u_i \in S$ can learn all edges between their neighbors in one round and then compute $\mathrm{CapDS}(v, C_v)$ locally without communicating. □

Computing a MIS: As discussed in detail in Sect. 2, the time T_{MIS} required to compute a maximal independent set depends on the underlying network graph model, as well as on the amount of information available to the nodes. The following is a brief summary.

- For general graphs, a MIS can be computed in expected $T_{\mathrm{MIS}} \in \mathrm{O}(\log n)$ rounds by using messages of size $\mathrm{O}(\log n)$ [1, 27].
- In unit disk graphs and general BIGs, there is a distributed algorithm that computes a MIS in time $T_{\mathrm{MIS}} \in \mathrm{O}(\log^* n)$ [19, 31].
- In unit disk graphs, if every node knows its own coordinates, then $T_{\mathrm{MIS}} \in \mathrm{O}(1)$.

In combination with Theorem 5.2, these bounds imply the required running time of our algorithm to compute a constant approximation to the capacitated dominating set problem in BIGs.

Message Size: As described above, Algorithm 3 requires sending messages of size $\mathrm{O}(\Delta \log n)$ (a neighbor v of a node $u \in S$ has to send a list of its at most Δ neighbors to u). It is straightforward to implement the algorithm in time $\mathrm{O}(T_{\mathrm{MIS}} \log n)$ by using messages of size only $\mathrm{O}(\log n)$ if the recursion depth of $\mathrm{CapDS}(v, C_v)$ can be bounded by $\mathrm{O}(\log n)$ and if we use a MIS algorithm which only requires messages of size $\mathrm{O}(\log n)$. This is possible by making the following simple adjustment. If the largest cluster C_t contains more than $2|C_v|/3$ elements, we can move $|C_v|/3$ of its elements to the cluster of v. This reduces the number of nodes in each recursive call by at least a factor of $2/3$.

Table 1 Best known upper bounds (UB) and lower bounds (LB) in both general graphs (GG) and unit disk graphs/bounded independence graphs (BIG). For general graphs, the table gives the fastest known algorithm to compute a polylogarithmic approximation to MDS/CapMDS. For UDG/BIGs, the fastest known algorithm for a constant approximation is given. All results given without citation have been developed in this paper

	GG–UB	GG–LB	UDG–UB	UDG–LB
MDS	$O(\log n)$ [20]	$\Omega(\sqrt{\frac{\log n}{\log n \log n}})$ [18]	$O(\log^* n)$ [19, 31]	$\Omega(\log^* n)$ [24]
CapMDS non-uniform	No local algorithm	$\tilde{\Omega}(\sqrt{n})$, $\tilde{\Omega}(\Delta)$	No local algorithm	$\tilde{\Omega}(\sqrt{n})$, $\tilde{\Omega}(\Delta)$
CapMDS uniform	No local algorithm	$\tilde{\Omega}(\sqrt{n})$, $\tilde{\Omega}(\Delta)$	$O(\log^* n)$	$\Omega(\log^* n)$ [24]
CapMDS $(1+\epsilon, 2)$-viol.	$O(\log^3 n + \frac{\log n}{\epsilon})$	$\Omega(\frac{1}{\epsilon}\sqrt{\frac{\log n}{\log n \log n}})$		

6 Summary and Conclusions

In this work, we have derived the first distributed approximation results on the *capacitated dominating set problem*, which naturally arises in numerous distributed application and networking scenarios.

It is interesting to compare our results with the best known results for the basic MDS problem, without capacity constraints. Table 1 summarizes the best known upper and lower bounds for MDS as well as for the different versions of CapMDS that we have studied in this paper. Our results show that even in simple geometric network graphs, the problem with non-uniform capacities is inherently *non-local*. That is—like 2-coloring a ring, or constructing an MST, for instance—CapMDS belongs to a class of distributed computing problems for which every distributed algorithm requires a running time that corresponds to the network diameter in the worst-case. Interestingly, if capacities are allowed to be violated by a factor of $1+\epsilon$, for an $\epsilon > 0$, the problem becomes much more local even in general graphs. The relaxed version of CapMDS belongs to the same class of distributed problems as MDS as well as problems like maximum matching or minimum vertex cover [18, 20]. For these problems, a logarithmic or polylogarithmic time complexity is both necessary and sufficient for achieving a polylogarithmic approximation ratio. In future work, it will be interesting to narrow the gap between upper and lower bounds and—more generally—to devise local approximation algorithms for other important network coordination problems.

References

1. Alon, N., Babai, L., Itai, A.: A fast and simple randomized parallel algorithm for the maximal independent set problem. J. Algorithms **7**(4), 567–583 (1986)
2. Alzoubi, K., Wan, P.-J., Frieder, O.: Message-optimal connected dominating sets in mobile ad hoc networks. In: Proc. of the 3rd ACM Int. Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC), EPFL Lausanne, Switzerland, pp. 157–164 (2002)
3. Awerbuch, B.: Complexity of network synchronization. J. ACM **32**(4), 804–823 (1985)

4. Bar-Ilan, J., Kortsarz, G., Peleg, D.: How to allocate network centers. J. Algorithms **15**(3), 385–415 (1993)
5. Bar-Ilan, J., Kortsarz, G., Peleg, D.: Generalized submodular cover problems and applications. Theor. Comput. Sci. **250**, 179–200 (2001)
6. Birk, Y., Keidar, I., Liss, L., Schuster, A., Wolff, R.: Veracity radius—capturing the locality of distributed computations. In: Proc. of the 25th Annual ACM Symposium on Principles of Distributed Computing (PODC) (2006)
7. Chan, H., Luk, M., Perrig, A.: Using clustering information for sensor network localization. In: Proc. of the 1st Conference on Distributed Computing in Sensor Systems (DCOSS) (2005)
8. Dai, F., Wu, J.: An extended localized algorithm for connected dominating set formation in ad hoc wireless networks. IEEE Trans. Parallel Distrib. Syst. **15**(10), 902–920 (2004)
9. Deb, B., Nath, B.: On the node-scheduling approach to topology control in ad hoc networks. In: Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC), pp. 14–26 (2005)
10. Elkin, M.: Distributed approximation—a survey. ACM SIGACT News—Distrib. Comput. Column **35**(4), 40–57 (2004)
11. Gandhi, R., Parthasarathy, S.: Distributed algorithms for coloring and connected domination in wireless ad hoc networks. In: Foundations of Software Technology and Theoretical Computer Science (FSTTCS) (2004)
12. Gao, J., Guibas, L., Hershberger, J., Zhang, L., Zhu, A.: Discrete mobile centers. In: Proc. 17th Symposium on Computational Geometry (SCG), pp. 188–196 (2001)
13. Garey, M.R., Johnson, D.S.: Computers and Intractability, A Guide to the Theory of NP-Completeness. Freeman, New York (1979)
14. Gfeller, B., Vicari, E.: A randomized distributed algorithm for the maximal independent set problem in growth-bounded graphs. In: Proceedings of the 26th ACM Symposium on Principles of Distributed Computing (PODC), pp. 53–60 (2007)
15. Grandoni, F., Krönemann, J., Panconesi, A., Sozio, M.: Primal-dual based distributed algorithms for vertex cover with semi-hard capacities. In: Proc. of the 24th Annual ACM Symposium on Principles of Distributed Computing (PODC), pp. 118–125 (2005)
16. Heinzelman, W.R., Chandrakasan, A., Balakrishnan, H.: Energy-efficient communication protocol for wireless microsensor networks. In: Proceedings of the 33rd Hawaii International Conference on System Sciences (HICSS), p. 8020. IEEE Comput. Soc., Los Alamitos (2000)
17. Jia, L., Rajaraman, R., Suel, R.: An efficient distributed algorithm for constructing small dominating sets. In: Proc. of the 20th ACM Symposium on Principles of Distributed Computing (PODC), pp. 33–42 (2001)
18. Kuhn, F., Moscibroda, T., Wattenhofer, R.: What cannot be computed locally! In: Proceedings of 23rd ACM Symposium on Principles of Distributed Computing (PODC), pp. 300–309 (2004)
19. Kuhn, F., Moscibroda, T., Wattenhofer, R.: The locality of bounded growth. In: Proc. of the 24th ACM Symp. on Principles of Distributed Computing (PODC) (2005)
20. Kuhn, F., Moscibroda, T., Wattenhofer, R.: The price of being near-sighted. In: Proc. of the 17th ACM-SIAM Symposium on Discrete Algorithms (SODA) (2006)
21. Kuhn, F., Moscriboda, T., Nieberg, T., Wattenhofer, R.: Fast deterministic distributed maximal independent set computation on growth-bounded graphs. In: Proc. 19th Symposium on Distributed Computing (DISC) (2005)
22. Kuhn, F., Wattenhofer, R.: Constant-time distributed dominating set approximation. Distrib. Comput. J. **17**(4), 303–310 (2005)
23. Kutten, S., Peleg, D.: Fast distributed construction of small k-dominating sets and applications. J. Algorithms **28**, 40–66 (1998)
24. Lenzen, C., Wattenhofer, R.: Leveraging Linial's locality limit. In: Proc. of 22nd Symposium on Distributed Computing (DISC), Arcachon, France, September 2008
25. Linial, N.: Locality in distributed graph algorithms. SIAM J. Comput. **21**(1), 193–201 (1992)
26. Linial, N., Saks, M.: Low diameter graph decompositions. Combinatorica **13**(4), 441–454 (1993)
27. Luby, M.: A simple parallel algorithm for the maximal independent set problem. SIAM J. Comput. **15**, 1036–1053 (1986)
28. Moscibroda, T., Wattenhofer, R.: Maximal independent sets in radio networks. In: Proc. of the 23rd ACM Symp. on Principles of Distributed Computing (PODC) (2005)
29. Naor, M., Stockmeyer, L.: What can be computed locally? SIAM J. Comput. **24**(6), 1259–1277 (1995)

30. Raghavan, P., Thompson, C.D.: Randomized rounding: A technique for provably good algorithms and algorithmic proofs. Combinatorica **7**(4), 365–374 (1987)
31. Schneider, J., Wattenhofer, R.: A log-star distributed maximal independent set algorithm for growth-bounded graphs. In: Proceedings of the 27th ACM Symposium on Principles of Distributed Computing (PODC) (2008)
32. Wan, P., Alzoubi, K., Frieder, O.: Distributed construction of connected dominating set in wireless ad hoc networks. In: Proceedings of INFOCOM (2002)
33. Wang, Y., Wang, W., Li, X.-Y.: Distributed low-cost backbone formation for wireless ad hoc networks. In: Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC), pp. 2–13 (2005)
34. Wu, J., Li, H.: On calculating connected dominating set for efficient routing in ad hoc wireless networks. In: Proc. of the 3rd Int. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM), pp. 7–14 (1999)