Super-Fast Distributed Algorithms for Metric Facility Location

Andrew Berns, James Hegeman, and Sriram V. Pemmaraju *

Department of Computer Science
The University of Iowa
Iowa City, Iowa 52242-1419, USA
[andrew-berns,james-hegeman,sriram-pemmaraju]@uiowa.edu

Abstract. This paper presents a distributed $O(1)$-approximation algorithm in the $\mathcal{CONGEST}$ model for the metric facility location problem on a size-n clique network that has an expected running time of $O(\log\log n \cdot \log^* n)$ rounds. Though metric facility location has been considered by a number of researchers in low-diameter settings, this is the first sub-logarithmic-round algorithm for the problem that yields an $O(1)$-approximation in the setting of non-uniform facility opening costs. Since the facility location problem is specified by $\Omega(n^2)$ bits of information, any fast solution in the $\mathcal{CONGEST}$ model must be truly distributed. Our paper makes three main technical contributions. First, we show a new lower bound for metric facility location. Next, we demonstrate a reduction of the distributed metric facility location problem to the problem of computing an $O(1)$-ruling set of an appropriate spanning subgraph. Finally, we present a sub-logarithmic-round (in expectation) algorithm for computing a 2-ruling set in a spanning subgraph of a clique. Our algorithm accomplishes this by using a combination of randomized and deterministic sparsification.

1 Introduction

This paper explores the design of "super-fast" distributed algorithms in settings in which bandwidth constraints impose severe restrictions on the volume of information that can quickly reach an individual node. As a starting point for our exploration, we consider networks of diameter one (i.e., cliques) so as to focus on bandwidth constraints only and avoid penalties imposed by network distance between nodes. We assume the standard $\mathcal{CONGEST}$ model [19], which is a synchronous message-passing model in which each node in a size-n network can send a message of size $O(\log n)$ along each incident communication link in each round. By "super-fast" algorithms we mean algorithms whose running time is strictly sub-logarithmic, in any sense - deterministic, in expectation, or with high probability (w.h.p.). Several researchers have previously considered the design of such "super-fast" algorithms; see [10,12,18] for good examples of relevant

* This work is supported in part by National Science Foundation grant CCF 0915543

concepts. The working hypothesis is that in low-diameter (communication) settings, where congestion, rather than network distance, is the main bottleneck, we should be able to design algorithms that are much faster than algorithms for "local" problems in high-diameter settings.

The focus of this paper is the *distributed facility location* problem, which has been considered by several researchers [6,14,16,17] in low-diameter settings. We first describe the sequential version of the problem. The input to the facility location problem consists of a set of *facilities* $\mathcal{F} = \{x_1, x_2, \ldots, x_m\}$, a set of *clients* $\mathcal{C} = \{y_1, y_2, \ldots, y_n\}$, an *opening cost* f_i associated with each facility x_i, and a *connection cost* $D(x_i, y_j)$ between each facility x_i and client y_j. The goal is to find a subset $F \subseteq \mathcal{F}$ of facilities to *open* so as to minimize the facility opening cost plus connection costs, i.e.,

$$FacLoc(F) := \sum_{x_i \in F} f_i + \sum_{y_j \in \mathcal{C}} D(F, y_j)$$

where $D(F, y_j) := \min_{x_i \in F} D(x_i, y_j)$. Facility location is an old and well-studied problem in operations research [1,3,4,8,21] that arises in contexts such as locating hospitals in a city or locating distribution centers in a region.

The *metric facility location* problem is an important special case of facility location in which the connection costs satisfy the following "triangle inequality:" for any $x_i, x_{i'} \in \mathcal{F}$ and $y_j, y_{j'} \in \mathcal{C}$, $D(x_i, y_j) + D(y_j, x_{i'}) + D(x_{i'}, y_{j'}) \geq D(x_i, y_{j'})$. The facility location problem, even in its metric version, is NP-complete and finding approximation algorithms for the problem has been a fertile area of research. A series of constant-factor approximation algorithms have been proposed for the metric facility location problem, with a steady improvement in the constant specifying the approximation factor. See [11] for a recent 1.488-approximation algorithm. This result is near-optimal because it is known [7] that the metric facility location problem has no polynomial-time algorithm yielding an approximation guarantee better than 1.463 unless $NP \subseteq DTIME(n^{O(\log \log n)})$. For non-metric facility location, a simple greedy algorithm yields an $O(\log n)$-approximation, and this is also optimal (to within a constant factor) because it is easy to show that the problem is at least as hard as set cover.

More recently, the facility location problem has also been used as an abstraction for the problem of locating resources in wireless networks [5,15]. Motivated by this application, several researchers have considered the facility location problem in a distributed setting. In [14,16,17], the underlying communication network is a complete bipartite graph with $\mathcal{F}$ and $\mathcal{C}$ forming the bipartition. At the beginning of the algorithm, each node, whether it is a facility or a client, has knowledge of the connection costs between itself and all nodes in the other part. In addition, the facilities know their opening costs. In [6], the underlying communication network is a clique. Each node in the clique may choose to open as a facility, and each node that does not open will connect to an open facility. Note that all of the aforementioned work assumes the $\mathcal{CONGEST}$ model of distributed computation. The facility location problem considered in [15] assumes that the underlying communication network is a *unit disk graph* (UDG), and

also considers the $\mathcal{LOCAL}$ model. While such a network can be of high diameter relative to the number of nodes in the network, this paper [15] reduces the UDG facility location problem to a low-diameter facility location-type problem and uses this in the eventual solution.

None of the prior papers, however, achieve near-optimal approximation (i.e., constant-factor in the case of metric facility location and $O(\log n)$ in the case of non-metric facility location) in *sub-logarithmic* rounds. While [6] does present a *constant-round*, constant-factor approximation to metric facility location on a clique, it is only for the special case of *uniform* metric facility location, i.e., when all facility opening costs are identical. The question that drives this paper, then, is the following: Can we achieve a distributed constant-factor approximation algorithm for the metric facility location problem in the clique setting in strictly sub-logarithmic time? One can ask similar questions in the bipartite setting and for non-metric facility location as well, but as a first step we focus on the metric version of the facility location problem on a clique.

Distributed facility location is challenging even in low-diameter settings because the input consists of $\Theta(m \cdot n + m)$ pieces of information, distributed across the network, which cannot quickly be delivered to a single node (or even a small number of nodes) due to the bandwidth constraints of the $\mathcal{CONGEST}$ model. Therefore, any fast distributed algorithm for the problem must be truly distributed and needs to take advantage of the available bandwidth and of structural properties of approximate solutions. Also worth noting is that even though tight lower bounds on the running times of distributed approximation algorithms have been established [9], none of these bounds extend to low-diameter settings.

2 Results

Main result. The main result of this paper is an $O(1)$-approximation algorithm for metric facility location on a clique which has an expected running time of $O(\log\log n \cdot \log^* n)$ rounds. If the metric satisfies additional properties (e.g., it has constant doubling dimension), then we obtain an $O(\log^* n)$-round $O(1)$-approximation to the problem. Our results are achieved via a combination of techniques that include (i) a new lower bound for the optimal cost of metric facility location and (ii) a sparsification technique combining randomization with a deterministic subroutine that repeatedly leverages the available bandwidth to process sparse subgraphs.

2.1 Overview of Technical Contributions

We start by describing the distributed facility location problem on a clique, as in [6]. Let (P, D) be a discrete metric space with point set $P = \{p_1, p_2, \ldots, p_n\}$. Let f_i be the opening cost of p_i. Now view the metric space (P, D) as a completely connected size-n network $C = (P, E)$ with each point p_i represented by a node, which we will also call p_i. Each node p_i knows f_i and the connection costs (distances) $D(p_i, p_j)$ for all $p_j \in P$. The problem is to design a distributed

algorithm that runs on C in the $\mathcal{CONGEST}$ model and produces a subset $X \subseteq P$ such that each node $p_i \in X$ opens and provides services as a facility and each node $p_i \notin X$ connects to the nearest open node. The goal is to guarantee that $FacLoc(X) \leq \alpha \cdot OPT$, where OPT is the cost of an optimal solution to the given instance of facility location and α is some constant. We call this the CliqueFacLoc problem. Of course, we also want our algorithm to be "super-fast" (in some sense), i.e., terminate in $o(\log n)$ rounds.

Our paper makes three main technical contributions.

- **A new lower bound for metric facility location.** For $p \in P$, let $B(p, r)$ denote the set of points $q \in P$ satisfying $D(p, q) \leq r$. For each p_i, let r_i be the non-negative real number satisfying

$$\sum_{q \in B(p_i, r_i)} (r_i - D(p_i, q)) = f_i$$

  As observed by Mettu and Plaxton [13], r_i exists and is unique. Bădoiu et al. proved in [2] that $\sum_{i=1}^{n} r_i$ is a constant-factor approximation for OPT in the case of *uniform* facility opening costs. This fact plays a critical role in the design of the constant-round, constant-factor approximation algorithm of Gehweiler et al. [6] for the special case of CliqueFacLoc in which all facility opening costs are identical. However, the sum $\sum_{i=1}^{n} r_i$ can be arbitrarily large compared to OPT when the f_i's are allowed to vary. (Consider an example consisting of only two nodes, one of whose opening costs is large in comparison to the other and to the distance between them.) Therefore, we apply the idempotent transformation

$$r_i \to \bar{r}_i = \min_{1 \leq j \leq n} \{D(p_i, p_j) + r_j\},$$

  and use $\bar{r}_i$ instead of r_i to derive a lower bound. Note that for any i, $\bar{r}_i \leq r_i$. We show later that $\sum_{i=1}^{n} \bar{r}_i$ bounds the optimal cost OPT from below (to within a constant factor) in the general case of non-uniform facility opening costs.
- **Reduction to an $O(1)$-ruling set.** Our next contribution is an $O(1)$-round reduction of the distributed facility location problem on a clique to the problem of computing an $O(1)$-ruling set of a specific spanning subgraph. Let $C' = (P, E')$ be a spanning subgraph of C. A subset $Y \subseteq P$ is said to be *independent* if no two nodes in Y are neighbors in C'. An independent set Y is a *maximal independent set* (MIS) if no superset $Y' \supset Y$ is independent in C'. An independent set Y is β-ruling if every node in P is at most β hops along edges in C' from some node in Y. Clearly, an MIS is a 1-ruling set. We describe an algorithm that approximates distributed facility location on a clique by first computing a spanning collection of subgraphs $C_1, C_2, C_3, \ldots$ in $O(1)$ rounds. Then we show that a solution to the facility location problem (i.e., a set of nodes to open) can be obtained by computing a β-ruling set for each of the subgraphs C_j, $j \geq 1$, and combining the solutions in a certain

way. We show that combining the β-ruling sets can also be done in $O(1)$ rounds. The parameter β affects the approximation factor of the computed solution and enforcing $\beta = O(1)$ ensures that the solution to facility location is an $O(1)$-approximation.
– **An $O(1)$-ruling set via a combination of randomized and deterministic sparsification.** We present an expected-$O(\log\log n \cdot \log^* n)$-round algorithm for computing a 2-ruling set of a given spanning subgraph C' of a clique C. We start by describing a deterministic "subroutine" that takes a subset $Z \subseteq P$ as input and computes an MIS of $C'[Z]$ (i.e., the subgraph of C' induced by Z) in c rounds if $C'[Z]$ has at most $c \cdot n$ edges. This is achieved via a simple load-balancing scheme that communicates the entire subgraph $C'[Z]$ to all nodes in c rounds. We then show how to use randomization repeatedly to peel off subgraphs with linearly many edges for processing by the aforementioned subroutine. In this manner, the entire graph C' can be processed using a number of calls, to the deterministic subroutine, which is $O(\log\log n \cdot \log^* n)$ in expectation.

3 Reduction to the $O(1)$-Ruling Set Problem

3.1 A New Lower Bound for Non-uniform Metric Facility Location

In this subsection we show that $\sum_{i=1}^{n} \bar{r}_i$ is a constant-factor lower bound to the optimal cost OPT. To facilitate this, we recall a definition from Mettu and Plaxton [13]. The $charge(\cdot,\cdot)$ of a node p_i with respect to a collection of (open) facilities X (also known as a *configuration*) is defined by

$$charge(p_i, X) = D(p_i, X) + \sum_{p_j \in X} \max\{0, r_j - D(p_j, p_i)\}$$

where $D(p_i, X) = \min_{p_j \in X} D(p_i, p_j)$. Mettu and Plaxton showed that the cost of a configuration X, $FacLoc(X)$, is precisely equal to the sum of the charges with respect to X, i.e. $\sum_{i=1}^{n} charge(p_i, X)$ [13].

The Mettu-Plaxton configuration F_{MP} is derived from the Mettu-Plaxton algorithm [13]. This algorithm, referred to as the MP-algorithm, is a sequential, greedy algorithm for facility location on a clique in which facilities open (sequentially) precisely when there is no already-open facility within distance $2 \cdot r_i$ [13]. Our algorithm borrows from the core ideas of the MP-algorithm.

The F_{MP} configuration was shown to have a cost at most three times OPT [13]. So for any configuration X,

$$FacLoc(X) \geq \frac{1}{3} FacLoc(F_{MP}) = \frac{1}{3} \sum_{i=1}^{n} charge(p_i, F_{MP})$$

We now present the following lemma, which relates $FacLoc(X)$ (for any X) to $\sum_{i=1}^{n} \bar{r}_i$.

Lemma 1. *$FacLoc(X) \geq (\sum_{i=1}^{n} \bar{r}_i)/6$ for any configuration X.*

3.2 Algorithm

We present our facility location algorithm in Algorithm 1. Our distributed algorithm consists of three stages. We use the notations $G[H]$ and $E[G]$ to refer to the subgraph induced by H and the edge set of G, respectively.

Algorithm 1 FACILITYLOCATION

Input: A discrete metric space of nodes (P, D), with opening costs;
a sparsity parameter s
Assumption: Each node knows its own opening cost and the distances from itself to other nodes
Output: A subset of nodes (a *configuration*) to be declared open

1. Each node p_i computes and broadcasts its value r_i; $r_0 := \min_i r_i$.
2. Each node computes a partition of the network into classes H_k with $3^k \cdot r_0 \leq r_j < 3^{k+1} \cdot r_0$ for $p_j \in H_k$.
3. Each node determines its neighbors within its own class; for $p_i, p_j \in H_k$, $(p_i, p_j) \in E[G[H_k]]$ if and only if $D(p_i, p_j) \leq r_i + r_j$.
4. All nodes now use procedure RULINGSET($\bigcup_k G[H_k]$, s) to determine, for each k, a sparse set $T_k \subseteq H_k$.
5. Each node broadcasts its membership status with respect to the sparse set T_k of its own class.
6. A node $p_i \in H_k$ declares itself to be open if:
   (i) p_i is a member of the sparse set $T_k \subseteq H_k$, and
   (ii) There is no node p_j belonging to a class $H_{k'}$, with $k' < k$, such that $D(p_i, p_j) \leq 2r_i$.
7. Each node broadcasts its status (open or not), and nodes connect.

Stage 1 (Steps 1-2). Each node knows its own opening cost and the distance to other nodes, so node p_i computes r_i and broadcasts that value to all others. Once this is complete, each node knows all of the r_i values. Next, every node computes a partition of the network into groups whose r_i values vary by at most a factor of 3 (Step 2). Specifically, let $r_0 := \min_{1 \leq j \leq n}\{r_j\}$, and define the class H_k to be the set of nodes p_i such that $3^k \cdot r_0 \leq r_i < 3^{k+1} \cdot r_0$. Every node computes the class into which each node in the network, including itself, falls.

Stage 2 (Steps 3-5). We now focus our attention on class H_k. Suppose $p_i, p_j \in H_k$. We define p_i and p_j to be *adjacent* in class H_k if $D(p_i, p_j) \leq r_i + r_j$. Each node in H_k can determine its neighbors in H_k. We refer to the graph on nodes in H_k induced by this adjacency condition as $G[H_k]$. Next, consider the graph (on all n nodes) $\bigcup_k G[H_k]$ which is the union of all induced graphs $G[H_k]$. A sparse set of nodes in $\bigcup G[H_k]$ determines, for each k, a sparse set $T_k \subseteq H_k$. Therefore, apply procedure RULINGSET() to $\bigcup_k G[H_k]$. We describe a super-fast-expected-

time implementation of RulingSet() in Section 4. After a sparse set has been constructed for each class H_k, each node broadcasts its membership status.

Stage 3 (Steps 6-7). Finally, a node p_i in class H_k opens if (i) $p_i \in T_k$, and (ii) there is no node $p_j \in B(p_i, 2r_i)$ of a class $H_{k'}$ with $k' < k$. Open facilities declare themselves via broadcast, and every node connects to the nearest open facility.

3.3 Analysis

We show that our algorithm produces an $O(s)$-approximation to CliqueFacLoc in $O(\mathcal{T}(n,s))$ communication rounds, where $\mathcal{T}(n,s)$ is the running time (in rounds) of procedure RulingSet().

Communication rounds. Stage 1 requires exactly one round of communication, to broadcast r_i values. Stage 2 requires $O(\mathcal{T}(n,s))$ rounds to compute the s-sparse subsets $\{T_k\}_k$, and an additional round to broadcast membership status. Stage 3 requires one round, in order to inform others of a nodes decision to open or not. Thus, the running time of our algorithm in communication rounds is $O(\mathcal{T}(n,s))$.

Cost approximation. Let F be the set of nodes opened by our algorithm. We analyze $FacLoc(F)$ by bounding $charge(p_i, F)$ for each p_i. Recall that $FacLoc(F) = \sum_{i=1}^{n} charge(p_i, F)$. Since $charge(p_i, F)$ is the sum of two terms, $D(p_i, F)$ and $\sum_{p_j \in F} \max\{0, r_j - D(p_j, p_i)\}$, we bound each separately by a $O(s)$-multiple of $\bar{r}_i$.

Algorithm 2 RulingSet

Input: An undirected graph $G = (V, E)$; a sparsity parameter s
Assumptions: Each node has knowledge of its neighbors in G; each node can send a message to any other node (not just along edges of G)
Output: An independent s-ruling set $T \subseteq G$

The sparse subset $T_k \subseteq H_k$ has the property that for any node $p_i \in H_k$, $D(p_i, T_k) \leq 2 \cdot 3^{k+1} r_0 \cdot s$, where s is the sparsity parameter passed to procedure RulingSet(). Also, for no two members of T_k is the distance between them less than $2 \cdot 3^k r_0$. Note that here we are using distances from the metric D of (P, D).

Now, in our cost analysis, we consider a node $p_i \in H_k$. To bound $D(p_i, F)$, observe that either $p_i \in T_k$, or else there exists a node $p_j \in T_k$ such that $D(p_i, p_j) \leq 2 \cdot 3^{k+1} r_0 \cdot s \leq 6 r_i \cdot s$. Also, if a node $p_j \in T_k$ does not open, then there exists another node $p_{j'}$ in a class $H_{k'}$, with $k' < k$, such that $D(p_j, p_{j'}) \leq 2 r_j$.

We are now ready to bound the components of $charge(p_i, F)$.

Lemma 2. *For all i, $D(p_i, F) \leq (81s + 81) \cdot \bar{r}_i$.*

Lemma 3. *For all i, $\sum_{p_j \in F} \max\{0, r_j - D(p_j, p_i)\} \leq 9\bar{r}_i$.*

Combining the two previous lemmas gives

$$FacLoc(F) = \sum_{i=1}^{n} charge(p_i, F) = \sum_{i=1}^{n} \left[D(p_i, F) + \sum_{p_j \in F} \max\{0, r_j - D(p_j, p_i)\} \right]$$
$$\leq \sum_{i=1}^{n} \left[(81s + 81) \cdot \bar{r}_i + 9\bar{r}_i \right] \leq (81s + 90) \cdot \sum_{i=1}^{n} \bar{r}_i$$

Theorem 1. *Algorithm 1 (*FacilityLocation*) computes an $O(s)$-factor approximation to* CliqueFacLoc *in $O(\mathcal{T}(n, s))$ rounds, where $\mathcal{T}(n, s)$ is the running time of the* RulingSet*() procedure called with argument s.*

4 Computing a 2-Ruling Set

The facility location algorithm in Section 3 depends on being able to efficiently compute an independent β-ruling set, for small β, of an arbitrary spanning subgraph C' of a clique C. This section describes how to compute an (independent) 2-ruling set of C' in a number of rounds which is $O(\log \log n \cdot \log^* n)$ in expectation.

4.1 A Useful Subroutine

We first present a deterministic subroutine for efficiently computing a maximal independent set of a sparse, induced subgraph of C'. For a subset $M \subseteq P$, we use $C'[M]$ to denote the subgraph of C' induced by M and $E[M]$ and $e[M]$ to denote the set and number (respectively) of edges in $C'[M]$. The subroutine we present below computes an MIS of $C'[M]$ in $e[M]/n$ rounds. Later, we use this repeatedly in situations where $e[M] = O(n)$.

We assume that nodes in P have unique identifiers and can therefore be totally ordered according to these. Let $\rho_i \in \{0, 1, \ldots, n-1\}$ denote the rank of node p_i in this ordering. Imagine (temporarily) that edges are oriented from lower-rank nodes to higher-rank nodes and let $\mathcal{E}(p_i)$ denote the set of outgoing edges incident on p_i. Let d_i denote $|\mathcal{E}(p_i)|$, the outdegree of p_i, and let $D_i = \sum_{j:\rho_j < \rho_i} d_j$ denote the outdegree sum of lower-ranked nodes.

The subroutine shares the entire topology of $C'[M]$ with all nodes in the network. To do this efficiently, we map each edge $e \in E[M]$ to a node in P. Information about e will be sent to the node to which e is mapped. Each node will then broadcast information about all edges that have been mapped to it. See Algorithm 3.

Theorem 2. *Algorithm 3 computes an MIS L of $C'[M]$ in $\frac{e[M]}{n} + O(1)$ rounds.*

Algorithm 3 Deterministic MIS for Sparse Graphs

Input: A subset of nodes $M \subseteq P$
Output: An MIS L of $C'[M]$
Algorithm executed by node p_i

1. Broadcast ID.
2. Calculate and broadcast d_i.
3. Assign a distinct label $\ell(e)$ from $\{D_i, D_i + 1, \ldots, D_i + d_i - 1\}$ to each incident outgoing edge e.
4. Send each outgoing edge e to a node p_j with rank $\rho_j = (\ell(e) \mod n)$.
5. Broadcast all edges received in previous step, one per round.
6. Compute $C'[M]$ from received edges and use a deterministic algorithm to locally compute MIS L.

4.2 Algorithm

We are now ready to present an algorithm for computing a 2-ruling set of C' which is "super-fast" in expectation. We show that this algorithm has an expected running time of $O(\log\log n \cdot \log^* n)$ rounds. The algorithm proceeds in *Stages*. In Stage i, $i = 1, 2, \ldots$ we process nodes whose degrees (in graph C') lie in the range $[n^{1/2^i}, n^{1/2^{i-1}})$. At the end of Stage i, every node has degree less than $n^{1/2^i}$; thus the algorithm consists of $O(\log\log n)$ Stages.

Each Stage consists of *Phases*. Consider the Stage in which we process the set $S(d)$ of nodes whose degrees are in the range $[d, d^2)$. In each Phase of this Stage, $|S(d)|$ decreases. To understand the rate at which this occurs, consider the function $t(k)$ defined recursively by $t(0) = 1$, $t(k+1) = e^{\sqrt{t(k)}}$, for all $k \geq 0$. This is a rapidly-growing function that reaches n in $O(\log^* n)$ steps. At the beginning of Phase k, $|S(d)| \leq n/t(k)$, and as Phase k proceeds, $S(d)$ shrinks. Phase k ends when $|S(d)| \leq \frac{n}{t(k+1)}$ (loop starting in Line 6). Because of the rate at which $t(k)$ grows, each Stage consists of $O(\log^* n)$ Phases.

Each Phase consists of *Iterations*. Consider the Stage in which we process nodes with degrees in $[d, d^2)$, and then consider an Iteration in Phase k of this Stage. In this Iteration, nodes in $S(d)$ join a set M independently with probability $q = \sqrt{t(k)}/d$ (Line 8). Nodes not in $S(d)$ join M with probability $1/\sqrt{d}$ (Line 9). The probability q is set such that the expected number of edges in $C'[M]$ is bounded above by $2n$. Once the set M is picked, we use Algorithm 3 to process $C'[M]$ in $O(1)$ rounds, and then we delete M and its neighborhood $N(M)$ (Lines 10-12). This ends an Iteration. In expectation, only a constant number of Iterations are needed to complete a Phase. Because the size of $S(d)$ decreases during a Phase, we can raise q (Line 15) while still ensuring that the expected number of edges in $C'[M]$ is $\leq 2n$. Within a Stage, q is increased until it reaches $1/\sqrt{d}$. As well, by the time q reaches $1/\sqrt{d}$, $S(d)$ will have diminished such that $|E[C'[S(d)]]| = O(n)$. The $S(d)$ remnant can then be processed in $O(1)$ time (Lines 18-20) to finish the Stage. See Algorithm 4.

Algorithm 4 Super-Fast 2-Ruling Set

Input: A spanning subgraph C' of the clique C
Output: A 2-ruling set $T \subseteq P$ of C'

1. $i = 1$; $d := \sqrt{n}$ (equal to $n^{1/2^i}$); $T := \emptyset$
2. **while** $d > 10$ **do**
   Start of Stage i***:***
3. $k := 0$; $q := \frac{1}{d}$ (equal to $\sqrt{t(k)}/d$);
4. $S(d) := \{p \in P : deg_{C'}(p) \geq d\}$; $lastPhase := false$;
5. **while** $(true)$ **do**
   Start of Phase k***:***
6. **while** $(|S(d)| > \frac{n}{t(k+1)}$ **and** $\neg lastPhase)$
   or $(|S(d)| > \frac{n}{e^{\sqrt{d}}}$ **and** $lastPhase)$ **do**
   Start of Iteration
7. $M := \emptyset$
8. Add each $p \in S(d)$ to M with probability q.
9. Add each $p \in P \setminus S(d)$ to M with probability $\frac{1}{\sqrt{d}}$.
10. Compute an MIS L on M using Algorithm 3.
11. $T := T \cup L$
12. Remove $(M \cup N(M))$ from C'.
13. $S(d) := \{p \in P : deg_{C'}(p) \geq d\}$
    End of Iteration
14. **if** $lastPhase$ **then break**;
15. $q := \frac{\sqrt{t(k+1)}}{d}$; $k := k + 1$;
16. **if** $q > \frac{1}{\sqrt{d}}$ **then**
17. $q := \frac{1}{\sqrt{d}}$; $lastPhase := true$;
    End of Phase
18. $M := S(d)$; Compute an MIS L on M using Algorithm 3.
19. $T := T \cup L$
20. Remove $(M \cup N(M))$ from C'.
21. $d := n^{1/2^{i+1}}$; $i := i + 1$
    End of Stage
22. $M := C'$; Compute an MIS L on M using Algorithm 3.
23. $T := T \cup L$
24. Output T.

4.3 Analysis

Lemma 4. *Algorithm 4 computes a 2-ruling set of C'.*

Lemma 5. *For any $d \geq 0$, the smallest k for which $t(k) \geq d$ is $O(\log^* d)$.*

Lemma 6. *Consider Phase k in Stage i. Let $d = n^{1/2^i}$. Then the maximum degree (in C') of a node during Phase k is less than d^2. Furthermore, $|S(d)| \leq n/t(k)$.*

Lemma 7. *In any Iteration, the expected number of edges in the subgraph (of C') induced by M is bounded above by $2n$.*

Lemma 8. *Fix a Stage i, and suppose that $t(k) \leq d$. Then the expected number of Iterations (Lines 7-13) required in Phase k before $|S(d)| \leq \frac{n}{t(k+1)}$ is $O(1)$.*

Lemma 9. *Fix a Stage i, and suppose that $t(k) \leq d < t(k+1)$. Then the expected number of Iterations required in Phase $k+1$ before $|S(d)| \leq \frac{n}{e^{\sqrt{d}}}$ is $O(1)$, and at the end of Phase $k+1$, the number of edges in $C'[S(d)]$ is $O(n)$.*

Theorem 3. *Algorithm 4 computes a 2-ruling set of C' and has an expected running time of $O(\log\log n \cdot \log^* n)$ communication rounds.*

5 Conclusions

Using Algorithm 4 as a specific instance of the procedure RulingSet() for $s = 2$ and combining Theorems 1 and 3 leads us to Theorem 4. We also note that under special circumstances an $O(1)$-ruling set of a spanning subgraph of a clique can be computed even more quickly. For example, if the subgraph of C induced by the nodes in class H_k is growth-bounded for each k, then we can use the Schneider-Wattenhofer [20] result to compute an MIS for $G[H_k]$ in $O(\log^* n)$ rounds (in the $\mathcal{CONGEST}$ model). It is easy to see that if the metric space (P, D) is Euclidean with constant dimension or even has constant doubling dimension, H_k would be growth-bounded for each k.

Theorem 4. *There exists an algorithm that solves the* CliqueFacLoc *problem with an expected running time of $O(\log\log n \cdot \log^* n)$ communication rounds.*

Theorem 5. *The* CliqueFacLoc *problem can be solved in $O(\log^* n)$ rounds on a metric space of constant doubling dimension.*

References

1. Balinski, M.: On finding integer solutions to linear programs. In: Proceedings of IBM Scientific Computing Symposium on Combinatorial Problems. pp. 225—248 (1966)

2. Bădoiu, M., Czumaj, A., Indyk, P., Sohler, C.: Facility location in sublinear time. In: ICALP. pp. 866——877 (2005)
3. Cornuejols, G., Nemhouser, G., Wolsey, L.: Discrete Location Theory. Wiley (1990)
4. Eede, M.V., Hansen, P., Kaufman, L.: A plant and warehouse location problem. Operational Research Quarterly 28(3), 547——554 (1977)
5. Frank, C.: Algorithms for Sensor and Ad Hoc Networks. Springer (2007)
6. Gehweiler, J., Lammersen, C., Sohler, C.: A distributed O(1)-approximation algorithm for the uniform facility location problem. In: Proceedings of the eighteenth annual ACM symposium on Parallelism in algorithms and architectures. pp. 237–243. SPAA '06, ACM, ACM Press, New York, NY, USA (2006)
7. Guha, S., Khuller, S.: Greedy strikes back: Improved facility location algorithms. In: Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms. pp. 649——657. Society for Industrial and Applied Mathematics (1998)
8. Hamburger, M.J., Kuehn, A.A.: A heuristic program for locating warehouses. Management science 9(4), 643——666 (1963)
9. Kuhn, F., Moscibroda, T., Wattenhofer, R.: Local Computation: Lower and Upper Bounds. CoRR abs/1011.5470 (2010)
10. Lenzen, C., Wattenhofer, R.: Brief announcement: Exponential speed-up of local algorithms using non-local communication. In: Proceeding of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing. pp. 295——296. ACM (2010)
11. Li, S.: A 1.488-approximation algorithm for the uncapacitated facility location problem. In: Proceedings of the 38th international colloquium on automata, languages and programming. pp. 77——88. ICALP '11, Springer-Verlag, Berlin, Heidelberg (2011)
12. Lotker, Z., Patt-Shamir, B., Pavlov, E., Peleg, D.: Minimum-weight spanning tree construction in O(log log n) communication rounds. SIAM J. Comput. 35(1), 120——131 (2005)
13. Mettu, R.R., Plaxton, C.G.: The online median problem. SIAM J. Comput. 32(3), 816——832 (2003)
14. Moscibroda, T., Wattenhofer, R.: Facility location: distributed approximation. In: Proceedings of the twenty-fourth annual ACM symposium on Principles of distributed computing. pp. 108——117. ACM, ACM Press, New York, NY, USA (2005)
15. Pandit, S., Pemmaraju, S.V.: Finding facilities fast. Distributed Computing and Networking pp. 11——24 (2009)
16. Pandit, S., Pemmaraju, S.V.: Return of the primal-dual: distributed metric facility location. In: Proceedings of the 28th ACM symposium on Principles of distributed computing. pp. 180——189. PODC '09, ACM, ACM Press, New York, NY, USA (2009)
17. Pandit, S., Pemmaraju, S.V.: Rapid randomized pruning for fast greedy distributed algorithms. In: Proceedings of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing. pp. 325——334. ACM (2010)
18. Patt-Shamir, B., Teplitsky, M.: The round complexity of distributed sorting: extended abstract. In: PODC. pp. 249——256. ACM Press (2011)
19. Peleg, D.: Distributed computing: a locality-sensitive approach, vol. 5. Society for Industrial and Applied Mathematics (2000)
20. Schneider, J., Wattenhofer, R.: A log-star distributed maximal independent set algorithm for growth-bounded graphs. In: Proceedings of the twenty-seventh ACM symposium on Principles of distributed computing. pp. 35——44. ACM (2008)
21. Stollsteimer, J.F.: A working model for plant numbers and locations. Journal of Farm Economics 45(3), 631——645 (1963)