Revisiting Connected Dominating Sets: An Optimal Local Algorithm?*

Samir Khuller[1] and Sheng Yang[2]

1 **Dept. of Computer Science**
University of Maryland, College Park, USA
samir@cs.umd.edu

2 **Dept. of Computer Science**
University of Maryland, College Park, USA
styang@cs.umd.edu

Abstract

In this paper we consider the classical Connected Dominating Set (CDS) problem. Twenty years ago, Guha and Khuller developed two algorithms for this problem - a centralized greedy approach with an approximation guarantee of $H(\Delta)+2$, and a local greedy approach with an approximation guarantee of $2(H(\Delta)+1)$ (where $H()$ is the harmonic function, and Δ is the maximum degree in the graph). A local greedy algorithm uses significantly less information about the graph, and can be useful in a variety of contexts. However, a fundamental question remained - can we get a local greedy algorithm with the same performance guarantee as the global greedy algorithm without the penalty of the multiplicative factor of "2" in the approximation factor? In this paper, we answer that question in the affirmative.

1998 ACM Subject Classification G.2.2 Graph Algorithms

Keywords and phrases graph algorithms, approximation algorithms, dominating sets, local information algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2016.0

1 Introduction

A connected dominating set (CDS) in a graph is a subset of vertices that induces a connected subgraph, and is also a dominating set at the same time. A dominating set is a subset of vertices such that every node in the graph, is either in the dominating set, or adjacent to a node in the dominating set. Finding a minimum connected dominating set is NP-hard, and thus for the last twenty years, researchers have explored approximation algorithms for this problem starting with the work of Guha and Khuller [6][1]. One of the original motivations for the problem was in building a backbone for routing in the context of wireless ad hoc networks. Over time many other applications have been explored in the context of social networks and AI [2, 10, 3].

In their paper, Guha and Khuller [6] developed two simple approaches - the first one is a "local" approach, where we start from a single vertex in the solution, and incrementally (greedily) add neighboring nodes while maintaining a connected subset of nodes at all times. It is tempting to imagine that adding one node at a time, maintaining connectivity might work well. However, this does not work and there are instances where we might end up with

* This work is supported by NSF grant CCF 1217890.
[1] This work was recently awarded the ESA Test of Time Award.

a solution with $\Omega(n)$ nodes, while the optimum solution has only $O(1)$ nodes! Interestingly, a simple modification of this algorithm that explores a 2-hop neighborhood making greedy choices at each step that involves selecting *upto two* nodes at each step, works much better and they show that a $2(H(\Delta)+1)$ approximation can be obtained for this problem ($H(n) = \sum_{i=1}^{n} \frac{1}{i}$ is harmonic function, and Δ is the largest degree in the graph). The main benefits of the local algorithm are that no knowledge of the entire graph is needed at each step, and only the part of the "explored" graph suffices to select the next node. This may not be a useful for a static graph, as you can calculate it once and forever. But to get a CDS for a graph that changes dynamically, like social networks or wireless ad hoc networks, limiting the amount of information required can be both important and challenging.

They also developed an improved global algorithm that is again a greedy algorithm, and constructs a solution that does not maintain any connectivity property, and is only connected at the very end. This centralized greedy algorithm yields a bound of $H(\Delta)+O(1)$, eliminating the factor of 2, and gets us close to the lower bound on this problem of $(1-\epsilon)H(\Delta)$ due to set cover hardness [5]. Despite much interest (see book by Du and Wan [3]), the key question remained open for two decades - is there a local algorithm that gives us the same bound as the global algorithm?

In this paper, we answer this question in the affirmative. We develop a very simple local algorithm, and are able to show that it matches the bound of the global algorithm. Such a result is especially surprising due to its simplicity.

Connected Dominating Sets became a central topic in the context of wireless ad hoc networks, where the CDS acts as a routing backbone for packet routing. Often it is expensive to connect all the nodes, as the cost can become prohibitive, and in this case it is fine to connect *most* of the nodes (or a given fraction). Liu and Liang [9] formalized the problem of partial connected dominating sets (PCDS) and provided heuristics without performance guarantees. Avrachenkov et al. [1] defined the budgeted connected dominating set problem (BCDS) where we have a budget of k nodes and we wish to find a connected set of at most k nodes that maximizes the number of nodes it can cover. Inspired by applications in social networks, they developed practical heuristics using only local information. Khuller et al. [8] developed the first algorithms with theoretical guarantees for both these problems with approximation factors of $O(\ln \Delta)$ for PCDS and $\frac{1}{13}\left(1-\frac{1}{\epsilon}\right)$ for BCDS.

Extensions to node weighted versions were considered by Guha and Khuller [7] as well. Extensive research was subsequently done on this topic with the development of distributed algorithms [4], as well as for many special classes of graphs [3].

1.1 Our Contributions

Our results can be summarized as follows

- In Section 3, for the Connected Dominating Set problem, we obtain the first local information algorithm whose approximation ratio is within additive constant to global algorithm, i.e. $H(\Delta) + O(1)$. To be precise, our approximation guarantee is $H(2\Delta + 1) + 1$. This algorithm requires 2-hop local information (see Section 2 for definition).
- In Section 4, with 1-hop local information, we obtain an $H(\Delta) + 2\sqrt{H(\Delta)} + 1$ approximation algorithm. In addition to better approximation ratio, it also runs faster than the Randomized CDS algorithm [6] (Section 2.3), because it explores fewer nodes.

2 Background

We first review existing approaches [6, 2].

2.1 Global Algorithm for CDS

The global algorithm runs in two phases. Initially, all nodes are colored white. In the first phase, the algorithm iteratively adds a node to the solution, colors it black and all its adjacent white nodes gray. A *piece* is defined as a white node or a black connected component. A new node is chosen to be colored black to get maximum reduction in the number of pieces. This phase ends when no such node exists that can give non-zero reductions. At this time, there are no white nodes left. Intuitively speaking, black nodes are selected nodes, gray nodes are nodes that are dominated, i.e. adjacent to black nodes.

In the second phase, we start with a dominating set that consists of several black components that we need to connect. The connection is done by recursively connecting pairs of black components with a chain of vertices, until there is only one black component, which will be our final solution.

The approximation ratio for this algorithm is $H(\Delta)+2$, where $H(n)$ is harmonic function.

2.2 2-hop Local Information Algorithm for CDS

The same paper proposed another algorithm, using only local information. Instead of using information of the entire graph, it only relies on information within 2-hops to the nodes chosen in the solution. The formal definition of local information is as follows.

Before we define what local information is, we first define the distance between a node and a set of nodes.

▶ **Definition 1** (Distance)**.** In undirected graph, denote the distance between u and v in a graph as $d(u, v)$. It is the length of shortest path from u to v. $d(u, S)$ is defined to be $\min_{v \in S} d(u, v)$

We now define the local neighborhood of some node, or a set of nodes.

▶ **Definition 2** (Local Neighborhood)**.** Given a set of nodes S in graph G, the r-hop neighborhood around S is the induced subgraph of G containing all nodes v such that $d(v, S) \leq r$. We denote the r-hop neighborhood as $N^r(S)$. When there is no confusion, we use the same notation to denote the set $N^r(S) = \{v | d(v, S) \leq r\}$.

An algorithm with local information uses information only within the local neighborhood of the nodes it has chosen. To be specific, if at some step, the set of nodes that an algorithm has chosen is S, and we have r-hop local information, then we know the induced graph of $N^r(S)$, as well as the degree of nodes in $N^r(S)/N^{r-1}(S)$.

From an arbitrary starting node, nodes are added iteratively. For each loop in this algorithm, one chooses a node, or a node and one of its neighbors. This means we need knowledge of 2-hop neighborhood to maximize the number of newly covered nodes, which explains why it uses 2-hops of local information.

Algorithm 1: CDS with 2-hop Local Information

Data: Graph $G = (V, E)$
Result: A connected dominating set S
$s \leftarrow$ an arbitrary node;
$S \leftarrow \{s\}$;
while S *is not a dominating set* **do**
  $\bar{S} \leftarrow$ a node u in $N^1(S)$ or a node in $N^1(S)$ and one of its neighbors in $N^1(u)$ (which also lies in $N^2(S)$) that maximize the number of newly covered nodes;
  $S \leftarrow S \cup \bar{S}$;

The approximation ratio for this algorithm is $2(H(\Delta) + 1)$. We can improve it in practice by maximizing the number of newly covered nodes for each new node selected, which is used in [10], but the theoretical bound is the same.

2.3 1-hop Local Information Algorithm for CDS

Borgs et al. [2] first came up with this algorithm, which is based on the previous one. Instead of choosing a node or a pair of adjacent nodes greedily, it chooses only one gray node to maximize the number of newly covered nodes, and in addition selects one of the newly covered nodes uniformly at random. The maximization process only requires 1-hop neighborhood information, and we do not worry about the random node we are about to choose. Not only does this algorithm require less information, it runs much faster.

Algorithm 2: Randomized CDS with 1-hop Local Information

Data: Graph $G = (V, E)$
Result: A connected dominating set S
$s \leftarrow$ an arbitrary node;
$S \leftarrow \{s\}$;
while S *is not a dominating set* **do**
  $v \leftarrow$ a node in $N^1(S)$ that maximize the number of newly covered nodes;
  $u \leftarrow$ a uniformly randomly chosen node from $N^1(v) - N^1(S)$, i.e. the newly covered nodes;
  $S \leftarrow S \cup \{u, v\}$;

The approximation ratio for this algorithm is $2(H(\Delta) + 1)$.

2.4 Inspiration

Comparing the two algorithms using global information and local information, we find a gap of factor 2 in the approximation ratio: $2(H(\Delta)) + O(1)$ for a local algorithm versus $H(\Delta) + O(1)$ for global algorithm. This looks reasonable: we are always better off when offered more information. But is it really the case? Is this gap the price we paid for lack of information essential, or the same approximation ratio can be achieved regardless of limited information. Our answer is affirmative: with local information, we can still get $H(\Delta)+O(1)$ approximation. To be specific, an $H(2\Delta + 1) + 1$ approximation.

3 Improved 2-Hop Local Information Algorithm

3.1 Algorithm

S is initially any node. Iteratively add nodes within $N^2(S)$ to S. When new nodes are added to S, the 2-hop neighborhood of S expands, and we repeat this process. By the end of the process, we have colored all the nodes. Initially, all nodes are white. When a node is added to S, we color it black, and all its white neighbors gray. The selected black nodes will form components, and these components may merge when additional nodes are selected.

At each step, we look within a 2-hop neighborhood of S, i.e., among the nodes that are either gray or adjacent to gray nodes. Add the node v that maximizes $2w_v + c_v - 1$, where c_v is the number of different black components connected to v, and w_v is the number of white nodes in $N^1(v)$. Our knowledge of the graph is enriched when we add nodes to our solution. The algorithm ends when there is no v such that $2w_v + c_v - 1 > 0$. Note that if v was gray before the selection, c_v is guaranteed to be greater or equal to 1. If it was white, $c_v = 0$.

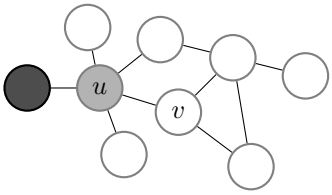

Figure 1 Consider u and v as potential choices. For node u, $c_u = 1$, as there is only adjacent component. Note that $w_u = 4$, it has four white neighbors, and itself is not white, hence $2 \cdot w_u + c_u - 1 = 8$. For node v, since it is white, there is no adjacent black component, so $c_v = 0$. Note w_v equals 3, since both node v itself and two of its neighbors are white. So the value is $2 \cdot w_v + c_v - 1 = 6 + 0 - 1 = 5$.

To make everything clear, here is the pseudo code.

Algorithm 3: CDS with 2-hop Local Information

Data: Graph $G = (V, E)$
Result: A connected dominating set S

```
s ← an arbitrary node;
S ← {s};
V̄ ← N²(S);
while ∃v ∈ V̄, s.t. 2w_v + c_v − 1 > 0 do
    v ← argmax_{v∈V̄} 2w_v + c_v − 1;
    S ← S ∪ {v};
    V̄ ← N¹(N¹(v));
    for all nodes in V, update c_v, w_v;
```

3.1.1 Correctness

First we prove that what we get is a dominating set. If not, then there exists a node that is not dominated, i.e. a white node u. But $2w_u + c_u - 1 = 2w_u + 0 - 1 \geq 2 - 1 > 0$, we would have added this node to our solution, a contradiction.

Next, we prove the following theorem that this dominating set is indeed connected.

▶ **Theorem 3.** *The solution returned by Algorithm 3 is connected.*

We have the following observation:

▶ Observation 4. When a node is added to our solution, it is within 2-hops of some black node.

It is true because otherwise, this node would be beyond our knowledge and would not be considered at this step. This ensures the following corollary:

▶ **Corollary 5.** *Each newly added node can be connected to existing components at the cost of at most one additional node.*

Proof of Theorem 3. We prove this by contradiction. Suppose the algorithm gives a dominating set that is not connected, i.e. has more than one component. Exactly one of these components will contain the starting node, call it starting component. Consider the first time when a node u outside the starting component joins our solution. According to Corollary 5, there must exist another node v that can join u to the starting component. Since u is disconnected from the starting component, v is not in our solution. But at the end of the algorithm, $2 \cdot w_v + c_v - 1 \geq 0 + 2 - 1 > 0$, which means that v can be added to our algorithm. This gives a contradiction. ◀

So our algorithm will indeed give a connected dominating set.

3.2 Analysis

We are going to use a charging scheme to charge the cost of each selected node and bound the total charge, which in turn bounds the number of nodes we select. Before that, we define uncharged black node as:

▶ **Definition 6.** An uncharged black node is a node that was charged once when it turned directly from white to black, and has not been charged afterwards.

For uncharged black nodes, the following lemma holds.

▶ **Lemma 7.** *Each component contains exactly one uncharged black node.*

The proof of this lemma will come later.

Recall in the algorithm, we select a node v that maximizes $2w_v + c_v - 1$, where w_v is the number of white nodes in $N^1(v)$, and c_v is the number of black components adjacent to v. We charge 1 for selecting this node, and split this charge into $2w_v + c_v - 1$ shares. For every white neighbor u of v, it takes two shares, i.e. gets $2/(2w_v + c_v - 1)$ charge. For all but one adjacent components, a share of charge is placed on the uncharged black node of this component (assuming the correctness of Lemma 7). Notice v may be white or gray before the selection. If it was white, it means that v is not adjacent to any black nodes, so $c_v = 0$, and it is charged one share, i.e. $1/(2w_v + c_v - 1)$. If it was gray, then nothing needs to be done. In conclusion, if the node was white, the number of shares equals $2(w_v - 1) + 0 + 1 = 2w_v + c_v - 1$; if the node was black, the number of shares equals $2w_v + (c_v - 1) = 2w_v + c_v - 1$. This means we are not over or under counting charges. A visual explanation is in Figure 2.

We now prove the correctness of Lemma 7.

Proof of Lemma 7. An uncharged black node comes into existence when a white node is chosen and added to our solution. According to the charging scheme, this node itself forms a component, and it has exactly one uncharged black node.

Figure 2 Consider the charging for u and v if they were selected at this step. For u, $2 \cdot w_u + c_u - 1 = 8$, each white neighbor of u gets 2 shares. For node v, $2 \cdot w_v + c_v - 1 = 6 + 0 - 1 = 5$. Each white neighbor gets 2 shares, but since node v goes from white directly to black, it only gets one share, and become an uncharged black node.

Components are connected when a gray node is chosen which connects several components. Since all but one component was charged, assuming all existing components have exactly one uncharged black node, the resulting component also has exactly one uncharged black node. A visual explanation is in Figure 3. ◀

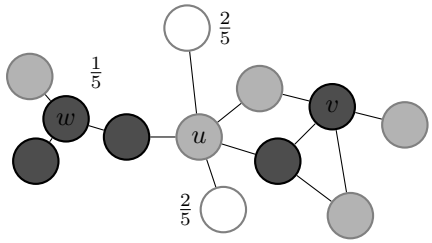

Figure 3 Suppose the black nodes are already chosen, and we are adding node u to the solution. There are two black components adjacent to u, so $c_u = 2$. Thus $2w_u + c_u - 1 = 4 + 2 - 1 = 5$. Each white neighbor of u gets 2 shares. For the two components, all but one component will get a share. This share is charged against the uncharged black node (node w for the left component and node v for the right component) of the component, which may not be the node adjacent to u. After charging, the two components are merged, and the merged component still has exactly one uncharged black node, i.e. node v. Note there is no charge against node u.

Everything prepared, we state the main theorem and prove it by bounding the total charge.

▶ **Theorem 8** (Main Theorem for 2-hop Local Information Algorithm). *The improved 2-hop local information algorithm gives $H(2\Delta + 1) + 1$ approximation.*

Proof of Theorem 8. Suppose the optimal solution is OPT, $|\text{OPT}| = k$, with vertices v_{OPT_1}, v_{OPT_2}, …, v_{OPT_k}. We partition all nodes into S_{OPT_1}, S_{OPT_2}, …, S_{OPT_k}. Without loss of generality, we reorder the nodes, and use i to denote vertex v_{OPT_i}. So S_i contains vertex i and its neighbors. Ties are broken arbitrarily as long as $i \in S_i$.

Without loss of generality, we assume that the charge in S_i changes at each step. To describe the total number of charges that nodes in S_i can receive, we define p_i^j as the following value for S_i in step j:

$$2 \cdot w_i^j + b_i^j - 1$$

where w_i^j is the number of white nodes in S_i at step j, b_i^j is the number of uncharged black nodes at step j in S_i. A symbol (e.g. w_i, b_i) with an upper script j, i.e. w_i^j, b_i^j, c_i^j, denotes the corresponding value at step j. $p_i^1 = 2\delta(i) + 1$ ($\delta(i)$ is the degree of node i). A white node can receive two charges (in fact all white nodes will receive two charges, except for one, which is the only uncharged black node when the algorithm ends). Either it will receive two charges when it first becomes gray, and never receive any more charge. Or it receives one charge when it becomes a uncharged black node, and receives another one to become a charged black node. We have the following lemma:

▶ **Lemma 9.** *When the total charge inside S_i changes, p_i^j decreases by at least one.*

Proof of Lemma 9. Total charge changes when some node in S_i receives charges. There can be three cases: the node was white, gray, or black. If this node was white before charging, either it is now gray, which means it receives two charges, or it is now black, meaning one charge. And p_i^j will decrease by 1 or 2. If it was gray, it must have been fully charged and will not receive any more charge. If it was black, it must have been an uncharged black node to receives one charge. In this case, p_i^j also decrease one. For all three cases, either there is no charge change in S_i, or there is a decrease on p_i^j of at least 1. ◀

We use p^j instead of p_i^j when there is no confusion. Notice $b_i^j \leq c_i^j$, since every uncharged black node corresponds to a component, but the uncharged black node may not be in S_i. Consider each step. Suppose the selected node is v . Since node i is also a valid choice, we have

$$2w_v^j + c_v^j - 1 \geq 2w_i^j + c_i^j - 1 \geq 2w_i^j + b_i^j - 1 = p^j$$

The number of charges that S_i receives at step j is $p^j - p^{j+1}$, so the total charge in this step is:

$$\frac{p^j - p^{j+1}}{2w_v^j + c_v^j - 1} \leq \frac{p^j - p^{j+1}}{p^j}$$

This holds when $2w_i^j + c_i^j - 1 > 0$, which is guaranteed to be true when $p_i^j > 0$. The inequality will break down when $w_v^j = 0$ and $c_v^j = 1$, i.e. when $2w_i^j + c_i^j - 1 = 0$. To fix it, we notice $\forall S_i, \exists k_i, s.t. p_i^{k_i} > 0, \forall t > k_i, p_i^t \leq 0$. Thus we can take out the last term and bound it separately. So the total charge until step k_i (including step k_i) is upper bounded by:

$$\begin{aligned}
\sum_{j=1}^{k_i} \frac{p^j - p^{j+1}}{p^j} &= \sum_{j=1}^{k_i} \sum_{t=p^{j+1}+1}^{p^j} \frac{1}{p^j} \\
&\leq \sum_{j=1}^{k_i-1} \sum_{t=p^{j+1}+1}^{p^j} \frac{1}{t} + \left(\sum_{p^{k_i+1}+1}^{\max\{p^{k_i+1}+1,1\}-1} 1 + \sum_{t=\max\{p^{k_i+1}+1,1\}}^{p^{k_i}} \frac{1}{t} \right) \\
&\leq \left(\sum_{j=1}^{k_i-1} \sum_{t=p^{j+1}+1}^{p^j} \frac{1}{t} + \sum_{t=\max\{p^{k_i+1}+1,1\}}^{p^{k_i}} \frac{1}{t} \right) + \sum_{p^{k_i+1}+1}^{\max\{p^{k_i+1}+1,1\}-1} 1 \\
&= \sum_{j=1}^{p^1} \frac{1}{j} + \sum_{p^{k_i+1}+1}^{\max\{p^{k_i+1}+1,1\}-1} 1 \\
&= H(p^1) + (0 - p_i^{k_i+1})
\end{aligned}$$

where $H(n) = \sum_{i=1}^{n} \frac{1}{i}$ is harmonic sum. The last equality uses the fact that $2w_v + c_v - 1 \geq -1$.

Using $p_i^{k_i+1} \leq 0$, $p_i^t \geq -1$, combined with our assumption that p_i^j decreases at each step, there is at most one more step before the whole algorithm stops. So the total amount of charge is upper bounded by:

$$(p_i^{k_i+1} - p_i^{k_i+2}) \cdot 1 \leq (p_i^{k_i+1} + 1)$$

Adding together, we have,

$$H(p^1) + (0 - p_i^{k_i+1}) + (p_i^{k_i+1} + 1) = H(p^1) + 1$$

As $p^1 = 2w_i + b_i - 1 = 2w_i - 1 \leq 2\Delta + 1$, the total charge in S_i is bounded by $H(2\Delta+1)+1$. Since there are $|OPT|$ different S_i, the total charge is bounded by $|OPT|(H(2\Delta+1)+1)$, which is also the upper bound for the number of nodes we choose. ◀

4 Improved 1-Hop Local Information Algorithm

4.1 Intuition

Recall that the algorithm by Borgs et al. [2], which selects a random neighbor when a gray node is chosen. It is too expensive to select a random node every time. If instead of picking a random neighbor every time, we can pick it with some probability p. The total approximation ratio will be (this is only an intuition, detailed proof is in Section 4.3) $(1+p)(\frac{1}{p}+H(\Delta))$ that can be minimized when $p = \frac{1}{\sqrt{H(\Delta)}}$. This works given the assumption that Δ is known before hand, which is not always the case.

4.2 Algorithm

Instead of using the largest degree in the graph, every time when calculating p, we use the largest degree in the explored graph. In another world, the largest degree in $N^1(S)$. Below is the pseudocode.

Algorithm 4: Improved Algorithm for CDS with 1-hop Local Information

Data: Graph $G = (V, E)$
Result: A connected dominating set S

```
s ← an arbitrary node;
S ← {s};
while S is not a dominating set do
    d ← the largest degree in N^1(S);
    p ← 1/√H(d);
    v ← a node in N^1(S) that maximizes the number of newly
      covered nodes;
    S ← S ∪ {v};
    if with probability p then
        u ← a node from the newly covered nodes uniformly at
          random;
        S ← S ∪ {u};
```

4.3 Analysis

Like the previous analysis, we do charging, and partition all the nodes into $S_1, S_2, \ldots, S_{OPT}$ in the same manner. We bound the total charge inside S_i, and the total number of nodes chosen in our solution that corresponds to these charges.

▶ **Theorem 10** (Main Theorem for 1-hop Local Information Algorithm)**.** *The improved 1-hop local information algorithm gives $H(\Delta) + 2\sqrt{H(\Delta)} + 1$ approximation.*

Proof of Theorem 10. When a node is selected in line 6 in Algorithm 4, we charge 1, and the charge is uniformly divided among all the newly covered nodes. In line 9, another node is chosen with probability $p = \frac{1}{\sqrt{H(d)}}$, where d is the largest degree we currently know. Thus whenever S_i receives charge c at some step, the expected number of nodes in our solution increases by $c(1+p)$. Note that this p changes over time.

The charging for each S_i is divided into two phases. The first phase ends when one of the nodes in S_i got chosen. In another word, we come to the second phase when v_i is visible. For the first phase, we model the charging process with the following problem.

▶ **Definition 11.** For $1 \le j \le n$, let X_j, Y_j be independent Bernoulli random variables with $E[X_j] = p_j \in [0,1], E[Y_j] = p'_j \in [0,1]$. Let T be the random variable denoting the smallest j such that $X_j Y_j = 1$ (or n if $X_j Y_j = 0$ for all j).

Here X_j is a random variable indicating whether a node in S_i was chosen in the j-th step. Y_j indicate whether a random neighbor was chosen at this step. We will prove the following theorem

▶ **Theorem 12.**

$$\mathbb{E}_T[\sum_{j=1}^{T}(X_j + X_jY_j)] \le 1 + \sqrt{H(\Delta)}$$

Proof. We prove it by induction. If $T = 0$, it is trivial. Suppose it holds for $T = t-1$, we prove it holds for t

$$\begin{aligned}
&\mathbb{E}_T\left[\sum_{j=1}^{T}(X_j + X_jY_j)\right] \\
=&p_1p'_1(1+1) + p_1(1-p'_1)\left(1 + \mathbb{E}_T\left[\sum_{j=2}^{T}(X_j + X_jY_j)\,|X_1 = 1, Y_1 = 0\right]\right) \\
&+ (1-p_1)\mathbb{E}_T\left[\sum_{j=2}^{T}(X_j + X_jY_j)\,|X_1 = 0\right] \\
=&2p_1p'_1 + p_1(1-p'_1) + p_1(1-p'_1)\mathbb{E}_T\left[\sum_{j=2}^{T}(X_j + X_jY_j)\right] \\
&+ (1-p_1)\mathbb{E}_T\left[\sum_{j=2}^{T}(X_j + X_jY_j)\right] \\
=&p_1 + p_1p'_1 + (1-p_1p'_1)\mathbb{E}_T\left[\sum_{j=2}^{T}(X_j + X_jY_j)\right]
\end{aligned}$$

$$
\begin{aligned}
&\leq p_1 + p_1 p_1' + (1 - p_1 p_1')(1 + \sqrt{H(\Delta)}) \\
&= p_1 + 1 + (1 - p_1 p_1')\sqrt{H(\Delta)} \\
&= 1 + \sqrt{H(\Delta)} + p_1(1 - p_1'\sqrt{H(\Delta)}) \\
&\leq 1 + \sqrt{H(\Delta)} + p_1(1 - \frac{1}{H(\Delta)} \cdot \sqrt{H(\Delta)}) \\
&= 1 + \sqrt{H(\Delta)}
\end{aligned}
$$

◀

The last inequality uses the fact that $p_j' = \frac{1}{\sqrt{H(\delta(v))}}$ for some node v, and $\delta(v) \leq \Delta$. So $p_j' \geq \frac{1}{H(\Delta)}$

As for the second phase, whenever S_i receives charge c, the expected number of nodes increases by $c(1+p)$, $p = \frac{1}{\sqrt{H(d)}}$, where d is the largest degree we currently know. Since $d_i = \delta(v_i)$ is already known, we have $d \geq d_i$. So $p = \frac{1}{\sqrt{H(d)}} \leq \frac{1}{\sqrt{H(d_i)}}$, which implies that the increase in expected number of nodes in S_i is bounded by $\frac{c}{\sqrt{(H(d_i))}}$

The total charge is bounded by $H(d_i)$. This can be done using the standard technique in set cover. Thus the total number of nodes chosen in phase 2 is bounded by

$$
\begin{aligned}
H(d_i)\,(1+p) &\leq H(d_i)\left(1 + \frac{1}{\sqrt{H(d_i)}}\right) = H(d_i) + \frac{H(d_i)}{\sqrt{H(d_i)}} = H(d_i) + \sqrt{H(d_i)} \\
&\leq \sqrt{H(\Delta)} + H(\Delta)
\end{aligned}
$$

Combining the charge from both phases, the expected number of nodes chosen in S_i is bounded by $H(\Delta) + 2\sqrt{H(\Delta)} + 1$. This directly means that expected the size of solution is bounded by $|OPT| \cdot (H(\Delta) + 2\sqrt{H(\Delta)} + 1)$, implying $H(\Delta) + 2\sqrt{H(\Delta)} + 1$ approximation. ◀

5 Future work

With only local information, we get the same approximation ratio as when we have global information, for connected dominating set problem. Is it also the case for other problems? Or does the lack of information prove to be a huge obstacle for designing algorithms?

Our first algorithm requires information within 2-hops. When only 1-hop local information is available, we cannot get the same result. Compared with previous result, the speed and approximation ratio is improved, i.e. $(H(\Delta) + 2\sqrt{H(\Delta)} + 1)$. But a gap still persists. Can we do better? Or is this the price we pay for lack of information?

References

1. Konstantin Avrachenkov, Prithwish Basu, Giovanni Neglia, Bernardete Ribeiro, and Don Towsley. Pay few, influence most: Online myopic network covering. In *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on*, pages 813–818. IEEE, 2014.
2. Christian Borgs, Michael Brautbar, Jennifer Chayes, Sanjeev Khanna, and Brendan Lucier. The power of local information in social networks. In *Internet and Network Economics*, pages 406–419. Springer, 2012.
3. Ding-Zhu Du and Peng-Jun Wan. *Connected dominating set: theory and applications*, volume 77. Springer Science & Business Media, 2012.

4 Devdatt Dubhashi, Alessandro Mei, Alessandro Panconesi, Jaikumar Radhakrishnan, and Aravind Srinivasan. Fast distributed algorithms for (weakly) connected dominating sets and linear-size skeletons. In *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 717–724. Society for Industrial and Applied Mathematics, 2003.

5 Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

6 Sudipto Guha and Samir Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20(4):374–387, 1998.

7 Sudipto Guha and Samir Khuller. Improved methods for approximating node weighted steiner trees and connected dominating sets. *Information and computation*, 150(1):57–74, 1999.

8 Samir Khuller, Manish Purohit, and Kanthi K Sarpatwar. Analyzing the optimal neighborhood: algorithms for budgeted and partial connected dominating set problems. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1702–1713. SIAM, 2014.

9 Yuzhen Liu and Weifa Liang. Approximate coverage in wireless sensor networks. In *Local Computer Networks, 2005. 30th Anniversary. The IEEE Conference on*, pages 68–75. IEEE, 2005.

10 Adish Singla, Eric Horvitz, Pushmeet Kohli, Ryen White, and Andreas Krause. Information gathering in networks via active exploration. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 981–988. AAAI Press, 2015.