Randomized algorithm for dominating set approximation problem

Piotr Kubaty

January 2023

1 Overview

Kuhn's and Wattenhofer's *Constant-Time Distributed Dominating Set Approximation* takes advantage of theoretical background of linear programming relaxation techniques to solve MDS problem in constant number of rounds, obtaining asymptotically nontrivial approximation. Let Δ be highest degree in graph $G = (V, E)$. The algorithm is parameterized by a parameter k. Algorithm consists of two parts. In part 1 it finds α approximation of fractional dominating set, where $\alpha = k(\Delta+1)^{\frac{2}{k}}$ when Δ is known (Algorithm 2 in referenced paper) or $\alpha = k((\Delta+1)^{\frac{2}{k}} + (\Delta+1)^{\frac{1}{k}}) \in O(k(\Delta+1)^{\frac{2}{k}})$ when Δ not known (Algorithm 3 in referenced paper). Part 1 is fully **deterministic**. In part 2 it uses **randomness** to transform fractional dominating into dominating set, obtaining dominating set S. **Expected** $\mathbb{E}[|S|] \leq (1 + \alpha \ln(\Delta+1)) \cdot |DS_{opt}|$.
For instance when $k = 6$ that obtains $O((\Delta+1)^{\frac{2}{6}} \ln(\Delta+1))$ - approximation to the dominating set problem and when $k = \Theta(\ln(\Delta+1))$ that gives $O((\ln(\Delta+1))^2)$ - approximation. Algorithm terminates in $O(k^2)$ rounds.

2 Algorithm

Algoritm 2 from reference is implemented in file
distributed-framework/graphs/mds/sync_kuhn_wattenhofer/sync_kuhn_wattenhofer_delta_known.go,

whereas **Algorithm 3** is implemented in
distributed-framework/graphs/mds/sync_kuhn_wattenhofer/sync_kuhn_wattenhofer_delta_unknown.go

all needed common utilities used in above files, as well as Algorithm 1 are implemented in
distributed-framework/graphs/mds/sync_kuhn_wattenhofer/common.go

Briefly, **algorithm 1** is a distributed algorithm, which for a given $i - th$ vertex uses its x_i domination fractional value of some $\alpha - approximation$ of fractional dominating set. It computes δ_2 defined as highest degree of any vertex in distance at most 2 from given vertex. There are two situations when vertex can be set to be in dominating set. **First situation** is being set randomly with probability $\min(1, x_i \cdot \ln(\delta_2 + 1))$. After that it can happen with some low probability, that given vertex is not in the dominating set and neither is any of its neighbors. In such case **second situation** occurs. Then we set this vertex to be in dominating set without any further communication with its neighbors.
Closed neighborhood (CN_i) of i-th vertex is the i-th vertex, as well as all adjacent vertices. **Algorithm 2** is also a distributed algorithm. Pseudocode:

```
$x_i := 0$
$status := covered$
for $l := k-1; k \geq 0; l--$
    for $m := k-1; m >= 0; m--$
        send status == uncovered to all neighbors
        $dynamicDelta$ := number of uncovered in closed neighborhood
        if $dynamicDelta > (\Delta+1)^{\frac{l}{k}}$ then $x_i = \max(x, (\Delta+1)^{\frac{-m}{k}})$
        send $x_i$ to all neighbors;
```

```
(9)     if $\sum x_j$ in closed neighborhood $\geq 1$ then status = covered
```

Algorithm 3 is a modification of Algorithm 2, substituting Δ (maximum degree in graph, as stated in the Overview) expressions with some local information. Besides that some helper information is passed in order to evaluate those expressions. That means about twice as much messages as in the Algorithm 2.

3 Correctness sketch

Algorithm 1. We want to prove $\mathbb{E}\left[|S|\right] \leq (1+\alpha\ln(\Delta+1))\cdot|DS_{opt}|$, when we are given α-approximation solution to fractional dominating set problem. Lets consider two random variables: X - number of vertices which are assigned to be in dominating in **first situation**. Because probability of a single vertex to be chosen in this case is $\min(1, x_i \cdot \ln(\delta_{2,i}+1))$ and because for each i $\delta_{2,i} \leq \Delta$, then

$$\mathbb{E}\left[X\right] \leq \sum x_i \cdot \ln(\delta_{2,i}+1) \leq \ln(\Delta+1)\sum x_i \leq \alpha\ln(\Delta+1))\cdot|FDS_{opt}| \leq \alpha\ln(\Delta+1))\cdot|DS_{opt}|,$$

where FDS_{opt} is minimal solution to fractional dominating set and DS_{opt} is minimal solution to dominating set problem. Define $\delta_{1,i}$ to be the greatest vertex degree in closed neighborhood of i. $\delta_{2,j} \geq \delta_{1,i}$ for all neighbours j of i.

$$\mathbb{E}\left[Y\right] = \sum_i \prod_{j\in CN_i} 1-(x_j \cdot \ln(\delta_{2,j}+1)) \leq \sum_i \prod_{j\in CN_i} 1-(x_j \cdot \ln(\delta_{1,i}+1)) \leq \sum_i (1-\frac{\sum_{j\in CN_i} x_j \cdot \ln(\delta_{1,i}+1)}{\delta_i+1})^{\delta_i+1}$$

The last inequality is the famous $AMGM$. Now using $\sum_{j\in CN_i} x_j \geq 1$ (α-approximation is a valid solution) and $1-x \leq e^{-x}$ we get

$$(1-\frac{\sum_{j\in CN_i} x_j \cdot \ln(\delta_{1,i}+1)}{\delta_i+1})^{\delta_i+1} \leq (1-\frac{\ln(\delta_{1,i}+1)}{\delta_i+1})^{\delta_i+1} \leq \frac{1}{\delta_{1,i}+1}.$$

$\sum_i \frac{1}{\delta_{1,i}+1} \leq |FDS_{opt}| \leq |DS_{opt}|$, because of weak duality theorem of linear programming (I will not elaborate here because its a different topic). Combining $\mathbb{E}\left[X\right] + \mathbb{E}\left[Y\right] = \mathbb{E}\left[X+Y\right]$ gives the result.
Algorithm 2. The idea here is to introduce helper semi-invariants. (Lemma 5.1) *dynamicDelta* at the beginning of the outer loop of the pseudocode above is always less or equal to $(\Delta+1)^{\frac{l+1}{k}}$. Proof is by induction. Lets observe that we start with $l = k-1$ and devrease it in each iteration. For $l = k-1$ the above invariant is trivially true. For subsequenent iterations it holds, because in previous iteration it was either already true or *dynamicDelta* was greater than $(\Delta+1)^{\frac{l+1}{k}}$. The latter implies that for $m = 0$ at line (7) in **previous outer iteration** the if condition was true and we set $x_i := 1$ and thus all its neighbours are covered at line (9). Hence in that case *dynamicDelta* is now 0.
Next in each iteration of inner loop we define $a(v_i)$ to be the number of uncovered nodes j in the closed neighborhood of v_i for which $dynamicDelta_j \geq (\Delta+1)^{\frac{l}{k}}$ if v_i is uncovered and 0 if v_i is covered. (Lemma 5.2) For each i $a(v_i) \leq (\Delta+1)^{\frac{m+1}{k}}$. Proof: assume contrary. Then for some i $a(v_i) > (\Delta+1)^{\frac{m+1}{k}}$. This means that in the **previous inner iteration** (inner index was m+1, l was the same as now) there were more than $(\Delta+1)^{\frac{m+1}{k}}$ vertices in closed neighborhood for which $dynamicDelta_j \geq (\Delta+1)^{\frac{l}{k}}$. It means that in line (7) corresponding x_js were set to be at least $(\Delta+1)^{\frac{-(m+1)}{k}}$. Therefore in line (9) $\sum_{j\in CN_i} x_j > 1$ and vertex i is covered, so $a(v_i) = 0$, contradiction.
Define helper variable z_i to be 0 at the beginning of each outer loop. Whenever x_i increases by ϵ, all uncovered x_j increase their z_j equally, so that their total increase in equal to ϵ(increase of x_i is distributed among $z_j s$) (Lemma 5.3) At the end of each iteration $z_i \leq \frac{1}{(\Delta+1)^{\frac{l-1}{k}}}$. I will give only brief idea. Observe from definition that only uncovered vertices increase their $z_i s$. Split the increase of z_i into two phases: The first phase consists of all inner-loop iterations where v_i remains uncovered. Then by $\sum_{j\in CN_i} x_j \leq 1$ at the end of this phase we obtain $z_i \leq \frac{\sum_{j\in CN_i} x_j}{(\Delta+1)^{\frac{l}{k}}} \leq \frac{1}{(\Delta+1)^{\frac{l}{k}}}$. Further, observe that in second phase z_i increases at most once: when x_i becomes covered. This phase makes use of lemma 5.2. That gives similar inequality and adding these two proves lemma.

(Theorem 5.4) Algorithm 2 computes feasible solution, which is $k(\Delta+1)^{\frac{2}{k}}$ approximation. Sketch: algo computes a feasible solution, because when $l, m = 0$, all uncovered yet vertices v_i have $x_i := 1$. Approximation part: in each iteration of the outer loop $\sum_{j \in CL_i} z_j \leq \frac{(\Delta+1)^{\frac{l+1}{k}}}{(\Delta+1)^{\frac{l-1}{k}}} = (\Delta+1)^{\frac{2}{k}}$. Numerator comes from Lemma 5.1 bound on $dynamicDelta_j$. $dynamicDelta_j$ is the number of uncovered nodes in neighbourhood, so precisely these, which can have their z_j increased. Denominator comes from lemma 5.3. Substitute $y_i = \frac{z_i}{(\Delta+1)^{\frac{2}{k}}}$. Then $\sum_{j \in CL_i} y_j \leq 1$, so ys form a feasible solution to dual problem in linear programming to fractional domianating set. By duality theorem in linear programming $\sum y_i \leq |FDS_{opt}| \leq |DS_{opt}|$. Finally there are k outer loop iterations, so we obtain inequality $\sum x_i \leq k \cdot (\Delta+1)^{\frac{2}{k}} \cdot |DS_{opt}|$.
Algorithm 3. Idea is very similar to Algorithm 2's. Lemmas 5.5, 5.6, 5.7 are analogs of 5.1, 5.2, 5.3, respectively. We make use of induction even more in them than in their analogs, in which sometimes some direct reasoning worked as well. Besides that observations are essentially the same. Advantage of local maximas of previously introduced variables is taken and the proof is even more technical.

4 Complexity

Algorithm 1 takes 3 communication rounds: 2 in order to evaluate δ_2 and 1 to send to all neighbors whether the vertex is in dominating set. Algorithms 2 and 3 consist of two nested loops, each iteration(round) consists of a constant number of communication rounds. That means $O(k^2)$ communication rounds in both cases. Each communication round consists of each vertex v sending and receiving $\deg(v)$ messages, which gives $2|E|$ messages in each communiacation round in total.

5 Reference

Kuhn, Wattenhofer - Constant-Time Distributed Dominating Set Approximation. Kuhn-Wattenhofer