University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
P. Bamberger, P. Schneider

Algorithm Theory
Sample Solution Exercise Sheet 10

Due: Tuesday, 11th of January, 2022, 4 pm

Exercise 1: Randomized Dominating Set *(20 Points)*

Let $G = (V, E)$ be an undirected graph. A set $D \subseteq V$ is called a *dominating set* if each node in V is either contained in D or adjacent to a node in D. The problem is to find a dominating set which is as small as possible (note that $D = V$ is trivially a dominating set). However, the problem of finding a minimum dominating set (or even a constant approximation) is NP-hard. In this exercise we present a randomized algorithm for d-regular graphs (i.e., graphs in which each node has exactly d neighbors) that computes a $O(\log n)$-approximation of a minimum dominating set.

Let $c > 0$.

Algorithm 1 `domset`(G)

1: $D = \emptyset$
2: Each node joins D independently with probability $p := \min\{1, \frac{(c+2)\ln n}{d+1}\}$
3: Each node that is neither in D nor has a neighbor in D joins D
4: **return** D

For simplicity, in all tasks you may assume that $\frac{(c+2)\cdot\ln n}{d+1} \leq 1$, i.e., that $p = \frac{(c+2)\cdot\ln n}{d+1}$.

(a) Explain the runtime of `domset`. *(1 Point)*

(b) Show that `domset` returns a dominating set with an expected size of $O\left(\frac{n \log n}{d}\right)$.

Hint: Use the inequality $(1 - x) \leq e^{-x}$. *(4 Points)*

(c) Show that after line 2 of `domset`, D has size $O\left(\frac{n \log n}{d}\right)$ with probability at least $1 - \frac{1}{n^{c+1}}$.

Hint: For $v \in V$*, let* X_v *be the random variable with* $X_v = 1$ *if* v *joins* D *in line 2 and* $X_v = 0$ *else. Now use Chernoff's Bound.* *(3 Points)*

(d) Show that with probability at least $1 - \frac{1}{n^{c+1}}$, no node joins D in line 3 of `domset`. *(3 Points)*

(e) Conclude that `domset` returns a dominating set of size $O\left(\frac{n \log n}{d}\right)$ with probability at least $1 - \frac{1}{n^c}$. *(1 Point)*

(f) Someone might now say: "Why not doing parts (c)-(e) like this: Let X_v be the random variable with $X_v = 1$ if v is in D (at the end of the algorithm) and $X_v = 0$ else. Then use Chernoff's Bound."

What would you respond?

Hint: Read the slide from the lecture about Chernoff Bounds carefully. *(1 Point)*

(g) Finally, show that `domset` computes an $\mathcal{O}(\log n)$-approximation of a minimum dominating set (i.e., $D \in \mathcal{O}(|D^*| \log n)$ where D^* is a minimum dominating set) with probability at least $1 - \frac{1}{n^c}$. *(3 Points)*

We now have shown that `domset` is a Monte Carlo algorithm for the problem "$O(\log n)$ minimum dominating set approximation". That is, `domset` has a fixed deterministic runtime and a probabilistic correctness guarantee.

(h) Describe a Las Vegas algorithm for "$O(\log n)$ minimum dominating set approximation". That is, your algorithm must always return the correct answer and its runtime must be polynomial in expectation *and* w.h.p. Prove that your algorithm has these properties. *(4 Points)*

Sample Solution

(a) Line 2 takes $\mathcal{O}(n)$ and line 3 $\mathcal{O}(nd)$ (for each node we must check its neighbors), so the runtime is $\mathcal{O}(nd)$.

(b) Line 3 ensures that D is a dominating set. Let $v \in V$.

$$\begin{aligned}\Pr(v \in D) &= \Pr(v \text{ joins } D \text{ in line 2}) + \Pr(v \text{ joins } D \text{ in line 3}) \\ &= \frac{(c+2)\ln n}{d+1} + \left(1 - \frac{(c+2)\ln n}{d+1}\right)^{d+1} \leq \frac{(c+2)\ln n}{d+1} + e^{-(c+2)\ln n} \\ &= \frac{(c+2)\ln n}{d+1} + \frac{1}{n^{c+2}}\end{aligned}$$

Now let X_v the random variable with $X_v = 1$ if $v \in D$ and $X_v = 0$ else.

We obtain

$$\begin{aligned}E[|D|] &= E[\sum_{v\in V} X_v] \overset{(*)}{=} \sum_{v\in V} E[X_v] = \sum_{v \in V} \Pr(v \in D) \leq n \cdot \left(\frac{(c+2)\ln n}{d+1} + \frac{1}{n^{c+2}}\right) \\ &= \frac{(c+2)n\ln n}{d+1} + \frac{1}{n^{c+1}} \leq \frac{(c+2)\cdot n\ln n}{d+1} + 1 = O\left(\frac{n\log n}{d}\right) .\end{aligned}$$

At $(*)$ we used linearity of expectation (which holds also for non-independent random variables).

(c) For repetition: If $X_1, \ldots, X_n$ is a sequence of independent 0-1 random variables, $X = \sum X_i$ and $\mu = E[X]$, then for any $\delta > 0$ we have

$$\Pr(X \geq (1+\delta)\mu) \leq e^{-\frac{\min\{\delta,\delta^2\}}{3}\mu} .$$

For each node v let X_v be the random variable with $X_v = 1$ if v joins D in line 2 and $X_v = 0$ else and let $X = \sum X_v$. We have $\Pr(X_v = 1) = \frac{(c+2)\ln n}{d+1}$ and hence $\mu = E[X] = \frac{(c+2)n\cdot\ln n}{d+1}$.

For $\delta = 3$ we obtain

$$\Pr(X \geq (1+\delta)\mu) \leq e^{-\mu} = e^{-\frac{(c+2)n\ln n}{d+1}} \leq e^{-(c+2)\ln n} = \frac{1}{n^{(c+2)}} \leq \frac{1}{n^{c+1}}$$

So with probability at least $1 - \frac{1}{n^{c+1}}$ we have $|D| \leq 4\mu = O\left(\frac{n\log n}{d}\right)$.

(d) v joins D in line 3 if neither v nor its d neighbors join D in line 2. We obtain

$$\Pr(v \text{ joins } D \text{ in line 3}) = \left(1 - \frac{(c+2)\ln n}{d+1}\right)^{d+1} \leq e^{-(c+2)\ln n} = \frac{1}{n^{c+2}}$$

A union bound over all nodes yields that the probability that some node joins D in line 3 is at most $\frac{1}{n^{c+1}}$.

(e) The probability that at least one of the conditions in (c) and (d) are not true is at most $\frac{2}{n^{c+1}} \leq \frac{1}{n^c}$.

(f) These random variables are not pairwise independent.

(g) For any dominating set D' we have $|D'| \geq \frac{n}{d+1}$. So by part (e), w.h.p., D is at most $O(\log n)$ times larger than a minimum dominating set.

(h) We repeat `domset` until it returns a set of size $O\left(\frac{n \log n}{d}\right)$. Running `domset` and checking the result takes $O(n+m)$. By part (e) we know that w.h.p., one run is sufficient.

For the expected runtime, we observe that the number of runs until the first success is geometrically distributed with parameter $p \geq 1 - \frac{1}{n^c}$. The expected number of trials is hence

$$\frac{1}{p} \leq \frac{1}{1 - \frac{1}{n^c}} \overset{n \to \infty}{\longrightarrow} 1 \ .$$

Therefore, our algorithm always outputs a correct solution (because we repeat `domset` until it does) and has a runtime of $O(n+m)$ in expectation and with high probability.