Algorithm HW#2

B02901178 江誠敏

June 3, 2017

1 Problem 1.

First we list some fact that we learned in class:

Lemma 1.

(1) The commute time $C_{u,v} \triangleq h_{u,v} + h_{v,u}$*, where* $h_{u,v}$ *is the expected time to walk to* v *from* u *could be calculated by* $C_{u,v} = 2mR_{u,v}$*, where* $R_{u,v}$ *is the effective resistance between node* u, v *defined in class.*

(2) For any connected graph, we have $C(G) \leq 2m(n-1)$ *where* $C(G) \triangleq \max C_u(G)$ *and* $C_u(G)$ *is the expected time to traverse from* u *to all other states. Tracing the proof given in class we also have a modified bound* $C(G) \leq 2mR_{u,v}$*.*

Now, for the lower bound, notice that $C_u(G) \geq \max_v h_{u,v}$ by definition. Also $2mR_{u,v} = C_{u,v} = h_{u,v} + h_{v,u}$ by (1) of lemma 1, which implies one of the $h_{u,v}, h_{v,u}$ is no less than $2mR_{u,v}/2 = mR_{u,v}$, thus

$$C(G) = \max C_u(G) \geq \max h_{u,v} \geq \max \{mR_{u,v}\} = mR(G)$$

For the upper bound, again by (1) of lemma 1,

$$h_{u,v} \leq C_{u,v} = 2mR_{u,v} \leq 2mR(G)$$

Now, for a random walk started from u, we shall calculate the probability of a vertex v that has not been visited after $16mR(G)$ steps. Recall Markov inequality:

Lemma 2 (Markov)**.** *If* X *is a random variable with* $\Pr\{X \geq 0\} = 1$*, then*

$$\Pr\{X \geq \alpha \, \mathbf{E}\, X\} \leq 1/\alpha$$

Since the expectation time to walk to v from u is $h_{u,v} \leq 2mR(G)$, the propability $\Pr\{v \text{ not visited}\} \leq 2mR(G)/(16mR(G)) = 2^{-3}$.

Now, after $16mR(G)$ steps, assume we end up in vertex u', and v has not been visited. We continue a walk of $16mR(G)$ steps from u', and again the probability of v not been visited in this walk is 2^{-3}. If we repeat the process for $\log_2 n$ times, we have

$$\Pr\{v \text{ not visited among these } \log_2 n \text{ walks}\} \leq (2^{-3})^{\log_2 n} = n^{-3}$$

Thus by union bound, if p is the probability that exists a vertex v not visited after $16mR(G)\log_2 n$ steps, then $p \leq n^{-3} \cdot n = n^{-2}$.

If we haven't visited all the vertex after $16mR(G)\log_2 n$ steps, and assume we are at vertex u. Forget about the previous walk and regard the walk afterward as a new random walk started from vertex u, then using the modified bound mentioned in (2) of lemma 1, the expected time to walk through all the vertex is bounded by $2mR(G)(n-1) \leq 2mn$, thus the expected time from the very begining is bounded by

$$\begin{aligned}
&(1-p)16mR(G)\log_2 n + p(16mR(G)\log_2 n + 2mR(G)n) \\
&\leq 16mR(G)\log_2 n + 2mR(G)/n \\
&\leq 32mR(G)\log_2 n
\end{aligned}$$

since $1/n \leq \log_2 n$.

If we could assume G is a simple graph, then another way to bound the expectation is to consider $32mR(G)\log_2 n$ step instead of $16mR(G)\log_2 n$ steps, and thus $p \leq n^{-3}$, so using the original bound in (2) of lemma 1,

$$\begin{aligned}
&(1-p)32mR(G)\log_2 n + p(32mR(G)\log_2 n + 2mn) \\
&\leq 16mR(G)\log_2 n + 2m/n^2 \\
&\leq 32mR(G)\log_2 n + 1
\end{aligned}$$

since $m \leq n^2/2$ now.

Collaborators: None.

2 Problem 2.

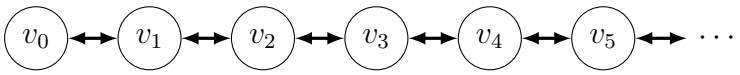
Follow the hint, first we prove a lemma:

Lemma 3. *Each flip in the algorithm decreases the difference with probability at least* $1/2$.

Proof. Without loss of generality, assume the clause is $(x_1 \vee x_2)$. Since the clause is not satisfied before flipping, both x_1 and x_2 are `False`, and in the satisfying assignment, at least one of them are `True`, which is choosed to flip with probability at least $1/2$. Other situation (e.g., $\neg x_1 \vee x_2$) follows similarly. □

Now, let u_k, $0 \leq k \leq n$ be the state such that the difference between the current assignment and the satisfying assignment is k, and $P(u_k, u_h)$ be the transition probability of two states, then u_k only transit to state u_{k+1}, u_{k-1} and $P(u_k, u_{k-1}) \geq 1/2$ by the lemma above. What we want is a bound of $E(u_k, u_0)$, the expected time to walk from u_k to u_0.

Consider the following undirected graph Markov chain:



Which $P(v_k, v_{k-1}) = 1/2 \leq P(u_k, u_{k-1})$ for $k \geq 1$, so we must have $E(v_k, v_0) \geq E(u_k, u_0)$. Now by (1) of lemma 1, $E(v_n, v_0) + E(v_0, v_n) = 2mR(v_k, v_0) = 2mk \leq 2n^2$ since the number of edge m is equal n by the fact that there are $n+1$ nodes. By symmetry, $E(v_0, v_n) = E(v_n, v_0)$, so $E(v_n, v_0) = n^2$. Also we have $E(v_n, v_0) \geq E(v_k, v_0)$ since the path from v_n to v_0 must go through v_k, so $E(v_n, v_0) = E(v_n, v_k) + E(v_k, v_0)$. Hence $E(u_k, u_0) \leq E(v_k, v_0) \leq E(v_n, v_0) \leq n^2$.

Using a similar argument in problem 1., consider a random walk with $2n^2 \log_2 n$ steps, divide into $\log_2 n$ section with $2n^2$ steps each. Each section could be regard as a random walk started at some vertex u_k, thus by Markov inequality, u_0 is not visited in this section has probability less then $n^2/(2n^2) = 1/2$, so the probability such that u_0 is not visited in $2n^2 \log_2 n$ steps is no more then $(1/2)^{\log_2 n} = 1/n$, thus the algorithm finds a satisfying assignment with probability at least $1 - 1/n$.

Collaborators: None.

3 Problem 3.

As hint, repeat A m times and get m estimations $a_1, \ldots, a_m$. Let a be the median of a_i, $R, L = (1 \pm \epsilon)\#(I)$, then if $\#\{i : a_i < L\} < m/2$ and $\#\{i : a_i > R\} < m/2$ both hold, we must have $L \leq a \leq R$. Since $\Pr\{L \leq a_i \leq R\} \geq 3/4$, we have $\Pr\{a_i < L\} \leq 1/4$. Now, let $X_i = 1$ if $a_i < L$, or else 0, then $\Pr\{X_i = 1\} \leq 1/4$ and each X_i are independent Bernoulli random variable. Let $X = \sum X_i$, we have

$$\Pr\Big\{\#\{i : a_i < L\} \geq \frac{m}{2}\Big\} = \Pr\Big\{X \geq \frac{m}{2}\Big\}$$

Recall Chernoff bound:

Lemma 4. *If $X_1, \ldots, X_m$ are independent Bernoulli random variable and $X = \sum X_i$, $\mu = \mathbf{E}\, X$, then* $\Pr\{X \geq \mu + \lambda\} \leq \exp\left(\frac{-2\lambda^2}{m}\right)$.

Now, $\mu \triangleq \mathbf{E}\, X \leq m/4$, so

$$\Pr\Big\{X \geq \frac{m}{2}\Big\} = \Pr\Big\{X \geq \mu + \frac{m}{2} - \mu\Big\} \leq \exp\left(\frac{-2(m/4)^2}{m}\right) \leq \exp\left(\frac{-m}{8}\right)$$

since $m/2 - \mu \geq m/4$. If we let $m = 8\log(2\delta)$ which is polynomial in $\log \delta$, then

$$\Pr\Big\{\#\{i : a_i < L\} \geq \frac{m}{2}\Big\} = \Pr\Big\{X \geq \frac{m}{2}\Big\} \leq \frac{1}{2\delta}$$

Similarly,

$$\Pr\Big\{\#\{i : a_i > R\} \geq \frac{m}{2}\Big\} \leq \frac{1}{2\delta}$$

By the union bound,

$$\begin{aligned}
&\Pr\{L \le a \le R\} \\
&= 1 - \Pr\Big\{\#\{i : a_i < L\} \ge \frac{m}{2} \text{ and } \#\{i : a_i > R\} \ge \frac{m}{2}\Big\} \\
&\le \frac{1}{2\delta} + \frac{1}{2\delta} = 1 - \frac{1}{\delta}
\end{aligned}$$

Since A is an algorithm with running time polynomial in n, ϵ^{-1}, and we repeat A m times, which is polynomial in $\log \delta^{-1}$, the overall running time is polynomial in $n, \epsilon^{-1}, \log \delta^{-1}$.

Collaborators: None.

4 Problem 4.

First we consider the probabilistic sampling problem:

Given an universe U, which each $x \in U$ appears with probability $p(x)$. Given $G \subseteq U$, Estimate $P \triangleq \sum_{x \in G} p(x)$.

This problem could be solved using a similar algorithm:

1. Sample n samples $X_1, \ldots, X_n$ by p.
2. Let $Y = \sum_i \mathbb{1}[X_i \in G]$, output $\hat{P} \triangleq Y/n$.

We have similar result to the original version.

Lemma 5. *If we pick $n \ge 3\log(2/\delta)/(\epsilon^2 P)$, then $\Pr\big\{|\hat{P} - P| \le \epsilon P\big\} \ge 1 - \delta$.*

Proof. Let $Y_i = \mathbb{1}[X_i \in G]$, then $Y = \sum Y_i$ and $\mathbf{E}\, Y_i = P$ implies that $\mathbf{E}\, Y = nP$. So

$$\begin{aligned}
\Pr\big\{|\hat{P} - P| \le \epsilon P\big\} &= \Pr\{|Y - \mathbf{E}\, Y| \le \epsilon\, \mathbf{E}\, Y\} \\
&= 1 - \Pr\{|Y - \mathbf{E}\, Y| \ge \epsilon\, \mathbf{E}\, Y\} \\
&\overset{(a)}{=} 1 - 2\exp\left(\frac{-\epsilon^2\, \mathbf{E}\, Y}{3}\right) \\
&\ge 1 - 2\exp(-\log(2/\delta)) = 1 - \delta
\end{aligned}$$

Where (a) is because of Chernoff bound. □

Now, as in the original DNF counting problem, let $\mathcal{I}$ be all the possible assignment (since there are n variables, $|\mathcal{I}| = 2^n$), $\mathcal{J} = \{1, \ldots, m\}$ be the indices of all clauses, and define $U \subseteq \mathcal{I} \times \mathcal{J}$ by $U \triangleq \{(\alpha, j) : \text{assignment } \alpha \text{ satisfies } j\text{th clause}\}$, Each $(\alpha, j) \in U$ has probability $p((\alpha, j)) = Cp(\alpha)$, where C is a normalizing factor, and $p(\alpha)$ is the probability of assignment α. That is, if α is the assignment that set $x_i : i \in A$ to `True`, then $p(\alpha) = \prod_{i \in A} p_i \cdot \prod_{i \notin A}(1 - p_i)$.

Define $U = \{(\alpha, j) : j \text{ is the smallest s.t. } (\alpha, j) \in U\}$, then the probability of the DNF be satisfied is $C^{-1} \sum_{(\alpha,j)\in U} p((\alpha, j))$. Notice that

$$1 = \sum_{(\alpha,i)\in U} p((\alpha, i)) = \sum_{(\alpha,j)\in G} \sum_{i:(\alpha,i)\in U} p((\alpha, i)) \leq \sum_{(\alpha,j)\in G} m \cdot p((\alpha, j))$$

So $q \triangleq \sum_{(\alpha,j)\in G} p((\alpha, j)) \geq 1/m$. By using the probabilistic sampling algorithm with lemma 5, we have an estimate $\hat{q}$ of q such that $\Pr\{|\hat{q} - q| \leq \epsilon q\} \geq 1 - \delta$, which only need $\mathcal{O}(m \log(\delta^{-1})/\epsilon^2$ samplings.

Now, we shall guarantee that sampling from U uniformly could be achieve in polynomial time. First define $P_j \triangleq \sum_{\alpha:(\alpha,j)\in U} p(\alpha)$, which is the sum of probability of those assignments which satisfy j-th clause. Define

$$A = \{i : x_i \text{ appears in the clause}\}$$
$$B = \{i : \neg x_i \text{ appears in the clause}\}$$
$$C = \{i : \text{both } x_i \text{ and } \neg x_i \text{ does not appear in the clause}\}$$

We could assume that $A \cap B = \varnothing$, or else the clause could never be satisfied. Then it is easy to see that if an assignment satisfy the clause, then $x_i : i \in A$ must be set to `True`, and $x_i : i \in B$ must be set to `False`, while the remains (i.e., $x_i : i \in C$) could be set to either one. Therefore the total probability is

$$P_j = \prod_{i\in A} p_i \cdot \prod_{i\in B} (1 - p_i)$$

which could be compute easily. Knowing P_i, $C = (\sum P_j)^{-1}$ could be calculated.

Finally, to sample (α, j) based on distribution $p((\alpha, j))$, we could first sample j by CP_j, and then sample an $(\alpha, j) \in U$ with probability $p((\alpha, j))/(CP_j) = p(\alpha)/P_j$. The last step is simple, just randomly set $x_i : i \in C$ to be `True` with probability p and `False` with probability $1 - p$ while each $x_i : i \in A$ set to `True` and $x_i : i \in B$ set to `False`.

The process mentioned above runs in polynomial of n, m, where n is the number of variables and m is the number of clauses, are all polynomial to the input size. The sampling algorithm runs in polynomial of $m, \log(\delta^{-1}), \epsilon^{-1}$, thus the overall algorithm is an RFTAS.

Collaborators: None.