Seminar on Sublinear Time Algorithms

Lecture 6

Juny 23, 2010

Lecturer: Robert Krauthgamer   Scribe by: Boris Temkin   Updated: July 20, 2010

1 Local partitioning of Planar Graphs

Let $G = (V, E)$ be an undirected graph with $|V| = n$ and maximal degree bounded by d.

Definition 1 *The set $V' \subseteq V$ is called a* **vertex cover** *of G if it incident (touches) to every edge in G. The* **vertex cover** *of minimal size is denoted by $VC(G)$.*

Our goal is following: Given a graph G as adjacency list, estimate $|\mathrm{VC}(G)|$ in sublinear time.

Theorem 1 (Hassidim-Kelner-Nguyen-Onak) *For every $\epsilon > 0$ and $d > 0$, there is an algorithm that given a planar graph $G = (V, E)$, with maximum degree $\leq d$ and $|V| = n$, approximates $|VC(G)|$ within additive ϵn (w.h.p) in time $T(\epsilon, d)$ independent of n.*

Proof Idea

- Fix a near optimal solution (vertex cover) C
- Sample $V^* \subseteq V$ uniformly at random such that $|V^*| = O(1/\epsilon^2)$
- Report $\frac{|V^* \cap C|}{|V^*|} \times n$

Outline of analysis

Firstly, we show that the algorithm estimates $|C|$ within additive ϵn w.h.p.
In order to do this, let $V^* = \left\{v_1, \ldots, v_{|V^*|}\right\}$, and $X_i = \mathbb{1}_{v_i \in C}$ and $X = \sum X_i$ (meaning $X \equiv |V^* \cap C|$).

$$\mathbb{E}[X] = \mathbb{E}\left[\sum X_i\right] = \sum \mathbb{E}[X_i] = |V^*| \times \frac{|C|}{n}$$
$$\mathrm{Var}[X] \leq \mathbb{E}[X] = |V^*| \times \frac{|C|}{n}$$

By Chebyshev inequality,

$$\Pr\left[\left|\frac{|V^* \cap C|}{|V^*|} \times n - |C|\right| \geq \epsilon n\right] = \Pr\left[\left|\frac{nX}{|V^*|} - |C|\right| \geq \epsilon n\right] = \Pr\left[\left|X - |V^*| \times \frac{|C|}{n}\right| \geq \epsilon |V^*|\right]$$
$$= \Pr\left[\left|X - \mathbb{E}[X]\right| \geq \epsilon |V^*|\right] \leq \frac{\mathrm{Var}[X]}{(\epsilon |V^*|)^2} \leq \frac{|V^*||C|}{n\epsilon^2 |V^*|^2} \leq \frac{1}{\epsilon^2 |V^*|} \leq \frac{1}{10} \quad (\text{as } |V^*| = O(1/\epsilon^2))$$

But we also need to show how to implement this in sublinear time.

Theorem 2 (Planar Separator Theorem (Lipton-Trojan)) *Let $G = (VE)$ be a planar graph. Then there exists a subset $S \subseteq V$ of size $O(\sqrt{|V|})$, such that in the graph $G \setminus S$, every connected component has size at most $\frac{1}{2}|V|$.*

(Observe that the theorem doesn't require bounded degree)
Remark The Planar Separator Theorem extends to every family of graphs, excluding a fixed minor. (by Alon-Seymour-Thomas)

Definition 2 *Define P to be a partition of the vertexes of G. I.e. $V = V_1 \uplus \ldots \uplus V_k$. We can think of it as $P\colon V \mapsto 2^V$.*

*P is called an (ϵ, k)-**partition** if $\forall v \in V$, $|P(v)| \leq k$ and $\forall i \neq j$, number of edges between V_i and V_j is at most $\epsilon|V|$.*

Corollary 3 *For every $\epsilon > 0$ and $d > 0$, there is $k^* = k^*(\epsilon, d)$, such that every planar graph of maximal degree $\leq d$ admits an (ϵ, k^*)-partition.*

Proof Exercise ■

Near optimal solution:

Also, we have to ensure, that $0 \leq |C| - |\mathrm{VC}(G)| \leq \epsilon n$.
Let C (near optimal solution) be defined as follows:

- Fix P as a (ϵ, k^*)-partition of G.
- Take an optimal solution for every part in P
- Add one endpoint for every cross-edge of P

Then it is easy to see that $|\mathrm{VC}(G)| \leq |C| \leq |\mathrm{VC}(G)| + \epsilon|V|$.

The algorithm for vertex cover actually uses an (ϵ, k)-partition P as follows:
For each $v \in V^*$ find an optimal vertex cover in $P(v)$ and check that v belongs to it. Also, take into account cross-edges.

So, we need to develop an oracle that given vertex v will return $P(v)$.

Oracle access for partition P:

Definition 3 *Let $G = (V, E)$ be a graph and $S \subseteq V$ be subset of vertices. Then define $e_{out(S)}$ to be the number of edges leaving S.*

Definition 4 *A set of vertices $S \subseteq V$ is called (k, δ)-isolated neighborhood of $v \in V$ if the following holds*

- $v \in S$

- $|S| = k$
- *S is connected in G*
- $e_{out(S)} \leq \delta|S|$

Consider the algorithm below for constructing P. The algorithm goes iteratively over the vertices of G in an arbitrary order, chooses a part of the graph around the current vertex (either an isolated neighborhood of the current vertex or just the current vertex alone) and removes this part from the graph G.

This description of the algorithm is sequential, but it can be implemented in sublinear time (providing oracle access to P) by letting the order of vertices (in said iteration) be according to a random permutation.

Algorithm 1 oracle for (ϵ, k)-partition

```
P ← ∅
π = (π_1, ..., π_|V|) ← random permutation of verices
for v according to the permutation π do
    if v ∈ V then
        if v has an (k, δ)-isolated neighborhood in subgraph induced on V then
            S ← (k, δ)-isolated neighborhood
        else
            S ← {v}
        end
        P ← P ∪ S
        V ← V \ S
    end
end
```

Lemma 4 *Let G' be an induced subgraph of G. Then the probability that a random vertex in G' doesn't have an $(k^*(\epsilon^2/800, d), \epsilon/20)$-isolated neighborhood is at most $\epsilon/20$*

Proof W.l.o.g. $G' = G$. Using Corollary 3, fix an $(\epsilon^2/800, k^*(\epsilon^2/800, d))$-partition P of G.

$$\mathop{\mathbb{E}}_{v \in V}\left[\frac{e_{\text{out}(P(v))}}{|P(v)|}\right] = \sum_{S \in P} \frac{|S|}{|V|} \cdot \frac{e_{\text{out}(S)}}{|S|} = \frac{1}{|V|} \sum_{S \in P} e_{\text{out}(S)} \leq \frac{2 \cdot \epsilon^2/800 \cdot |V|}{|V|} = \epsilon^2/400$$

By Markov inequality

$$\Pr_{v \in V}\left[\frac{e_{\text{out}(P(v))}}{|V|} \geq \epsilon/20\right] \leq \frac{\epsilon^2/400}{\epsilon/20} = \frac{\epsilon}{20}$$

■

Thus, for $1 - \epsilon/20$ fraction of the vertices $v \in V$, there is $P(v)$ for graph G' of current iteration such that the following holds

- $\frac{e_{\text{out}(P(v))}}{|P(v)|} < \epsilon/20$
- $P(v)$ is connected in G
- $v \in P(v)$
- $|P(v)| \leq k^*(\epsilon^2/800, d)$

Remark To compute $P(v)$ locally (i.e. without to compute whole P), we do following: Instead of using a random permutation, we generate for each vertex v, a random priority $r(v)$ on interval $[0, 1]$. We will generate it on demand and store already chosen values.
To compute $P(v)$, we recursively compute $P(w)$ for each vertex w such that $dist(w, v) < 2k$ and $r(w) < r(v)$.
If $v \in P(w)$ for some such w, then we set $P(v) = P(w)$.
Otherwise, we compute by exhaustive search an (k, δ)-isolated neighborhood of v, keeping in mind that all vertices $P(w)$ already removed from the graph.
If such neighborhood exists, we return it. Otherwise, we return $\{v\}$.
The detailed analysis is quite similar to maximum matching in lectures 2 and 3 and can be obtained from the courses' site.

Altogether , the following lemma give us over all running time for approximation of $|\mathrm{VC}\,(G)\,|$

Lemma 5 *Fix $\epsilon > 0$ and let $k = k^*(\epsilon^2/800, \delta = \epsilon/20)$. Then the algorithm above, computes with high probability an (ϵ, k)-partition. Moreover, if the oracle asked q non-adaptive queries, then with high probability its query complexity into G (and also its running time) is at most $q \cdot 2^{d^{O(k)}}$*