Sublinear approximation 17/02/2015

Thursday, February 19, 2015
4:41 PM

A detour from LP's into some recent work on using sampling to approximate in sublinear time

$o(n)$ where n is size of instance

Vertex Cover

Need linear time to read full input, so need to formalize query model to read parts of the input.

For VC, we use the following model.

- Can query $\deg(v)$ for any vertex v in unit time
- Can find the i'th nbr of a vertex v in unit time (can assume $i \in \deg(v)$)
- Can sample uniformly random vertex in unit time

We will assume that graph G is d-regular for simplicity (can actually work with avg. degree)

Definition: We say $\hat{Y}$ is an (α, β)-approximation of Y if $Y \leq \hat{Y} \leq \alpha \cdot Y + \beta$.

- Note that as Y becomes close to 0, an (α, β) approx becomes increasingly weaker than an $(\alpha, 0)$ approx

Recall: There is a linear time $(2,0)$-approx for vertex cover.

Pf sketch: Observe that for any maximal matching M, if $V(M)$ is the set of vertices covered by M and VC is size of optimal vertex cover, then:

$$\frac{1}{2} V(M) \leq VC \leq V(M)$$

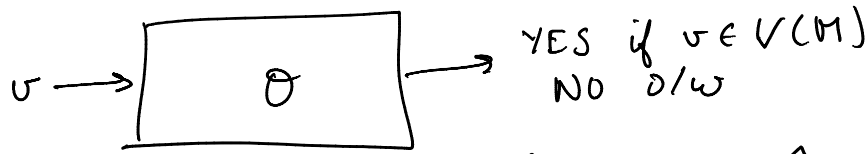
A maximal matching can be found by greedily selecting edges which are not adjacent to previously selected edges.

Today's main theorems: For any d-regular graph G, there is an ... that, with probability $\geq 2/3$, return a $(2, \varepsilon n)$ approx ... time.

Todays ... with probability $\geq 2/3$, return ...
algorithm that
of VC using $2^{O(d)} \cdot 1/\varepsilon^2$ queries of the above type.

Comment: A $O(d \cdot \text{poly}(1/\varepsilon))$ time algorithms is known.

Idea: We'll devise an oracle O that given a vertex v answers whether $v \in V(M)$ for some *fixed* hidden maximal matching M that it knows.



Given such an oracle O, finding $\hat{VC}$ s.t. $|\hat{VC} - VC| \leq \varepsilon n$ is easy.

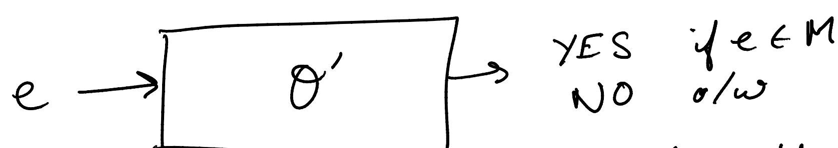
Claim: Let S be a random subset of V of size $O(1/\varepsilon^2 \lg 1/\delta)$. Let $Y(S) \subseteq S$ be the subset of S for which O answers YES. Then w/ prob $\geq 1-\delta$, $\left| \frac{|Y(S)|}{S} - \frac{|V(M)|}{V} \right| \leq \frac{\varepsilon}{2}$

Pf: Exercise in Chernoff bounds.

Claim: Using $O(1/\varepsilon^2 \lg 1/\delta)$ calls to O, we can find a $(2, \varepsilon n)$-approx of VC with probability $\geq 1-\delta$.

Pf: Think about it ...

We implement O by implementing another oracle O':



We can implement a call to O using d calls to O'.
$v \in V(M)$ iff $\exists e$ incident to v s.t. $e \in M$.

Implementing O'

How should O' choose its secret maximal matching M?
Suppose it ran the greedy algorithm by choosing edges

in a particular order.

Consider the following alg:

Let $l : e \to [0,1]$ be some injective map

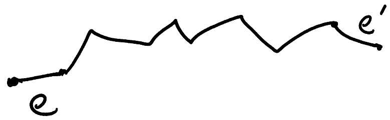
```
O'(e):
    For every edge e' sharing endpt with e:
        If l(e') > l(e):
            skip
        Else if l(e') < l(e):
            If O'(e') = YES
                return NO
    return YES
```

Clearly, we're simulating the greedy algorithm where edges are chosen according to increasing values of l. Also, note that l actually doesn't need to be known for all edges, just the edges that the oracle O' is called on. So, we will choose l "on the fly" and store the value of l for edges that have been queried.

l needs to be defined so that it's independent of the query sequence and is injective.

IDEA: let l be a <u>random</u> labeling.
Whenever we need $l(e)$, check if it's already set, otherwise set to be a random value in $[0,1]$.

All that remains is to bound the number of recursive calls to O' that are made for a single call to O'.



For a particular path P of length t, if e' is last edge of P,

$$\Pr[e' \text{ is recursively called}] = \frac{1}{t!}$$

$\mathbb{E}$[# of edges dist. t from e that are recursively called] $\leq \frac{(2d)^t}{t!}$

$\mathbb{E}$[# of recursive calls] $\leq \sum_{t=0} \frac{(2d)^t}{t!}$

$= e^{2d}$

So, total # of queries $= O(d \cdot e^{2d}/\varepsilon^2) = 2^{O(d)}/\varepsilon^2$.

In class, there was a suggestion by Palash Dey that O' be implemented as below (initialize M to $\emptyset$):

> $O'(e)$:
> If any adjacent edge of e has already been put in M, return NO
> Else put e in M and return YES

The issue with this proposal is that M is dependent on the sequence of oracle calls and is not fixed. So, it is not clear that we can estimate $V(M)$ by sampling random vertices, as we are no longer sampling from a fixed distribution.

Here is a concrete counter-example:

$n^{3/4}$ stars each with $n^{1/4}$ vertices

Whp. all $O(1/\varepsilon^2)$ vertices sampled will lie on the outside of distinct stars. The above algorithm will declare them all to be part of a maximal matching and our estimate of VC is going to be n.

On the other hand, $VC = n^{3/4}$