Streaming Algorithms

Meng-Tsung Tsai

04/24/2018

Reminder

Written Assignment #2 needs some fixes and the deadline is extended to Sunday midnight. You need to LaTex your solution and submit it on New E3 (https://e3new.nctu.edu.tw).

You are encouraged to discuss with your classmates, TA, or me. However, the writeup shall be your own.

We will announce Programming Assignment #1 this weekend. Please recall how to find articulation points by a DFS, on a RAM.

Fixes in Written Assignment #2

Problem #2

$O(1/c)$-pass → $O(c)$-pass

Problem #5

Let G be an arbitrary n-node m-edge connected undirected graph. Give a deterministic polynomial-time algorithm that can return n different cuts of G so that each partition the node set into two and the number of crossing edges is at least $(1-\varepsilon)m/2$ for an arbitrary $\varepsilon > 0$ if n is sufficiently large, as a function of ε.

References

- "A note on the Turán function of even cycles," Pikhurko (2010)

Spanners

Problem Definition

Input: Given an n-node m-edge undirected graph G and each edge e in G has weight $\omega(e)$.

Output: a sparse subgraph H of G so that $dis_H(u, v) \leq t \; dis_G(u, v)$ for some given t. Such an H is called t-spanner.

Goal: make H as sparse as possible.

It is NP-hard. Our goal is try to make H to be reasonably sparse.

Algorithm

```
Spanner(G = (V, E), t){

  Sort the edges in E by their weight in non-decreasing order.

  H ← (V, ∅);

  foreach (edge e){ // in the sorted order
    if (H ∪ {e} has a cycle of length ≤ t+1)
      discard e;
    else
      H ← H ∪ {e};
  }

  return H;
}
```

How to check whether $H \cup \{e\}$ has a cycle of length $\leq t+1$?

Algorithm

```
Spanner(G = (V, E), t){

  Sort the edges in E by their weight in non-decreasing order.

  H ← (V, ∅);

  foreach (edge e){ // in the sorted order
    if (H ∪ {e} has a cycle of length ≤ t+1)
      discard e;
    else
      H ← H ∪ {e};
  }

  return H;
}
```


H is a t-spanner.

Algorithm

Spanner(G = (V, E), t){

Sort the edges in E by their weight in non-decreasing order.

$H \leftarrow (V, \emptyset)$;

foreach (edge e){ // in the sorted order
  if ($H \cup \{e\}$ has a cycle of length $\leq t+1$)
    discard e;
  else
    $H \leftarrow H \cup \{e\}$;
}

return H;
}

H is C_{t+1}-free.

Erdős Even Circuit Theorem

Erdős Even Circuit Theorem

Theorem. Every n-node C_{2k}-free graph has $O(k\ n^{1+1/k})$ edges.

Erdős Even Circuit Theorem

Theorem. Every n-node C_{2k}-free graph has $O(k\ n^{1+1/k})$ edges.

Corollary. Spanner(G, t = (2k-1)) return an H with $O(k\ n^{1+1/k})$ edges.

Example. We can extract a subgraph H from the input graph G using $O(n^{1.5})$ space, and by which we can approximate the shortest path for every pair of nodes in G to within a factor of 3.

Why do Spanner(G, t) remove all cycles of length $\leq t+1$?
To use the even circuit theorem, we only need to remove all cycles of length = t+1.

Erdős Even Circuit Theorem

Theorem. Every n-node C_{2k}-free graph has $O(k\, n^{1+1/k})$ edges.

Corollary. Spanner(G, t = (2k-1)) return an H with $O(k\, n^{1+1/k})$ edges.

Example. We can extract a subgraph H from the input graph G using $O(n^{1.5})$ space, and by which we can approximate the shortest path for every pair of nodes in G to within a factor of 3.

Spanner(G, t) returns a subgraph of G whose girth is at least t+2.
Some people states this fact as large-girth graphs are sparse.

Proof Sketch

EC-Theorem (by Pikhurko)

Theorem 1. Every n-node C_{2k}-free graph G has fewer than

$$(k-1)n^{1+1/k} + 16(k-1)n \text{ edges.}$$

We prove this by contradiction. Our strategy:

1. Assume that G has at least $(k-1)n^{1+1/k} + 16(k-1)n$ edges.
2. Show that G must have a C_{2k}.

Let $\boldsymbol{\delta} = e(G)/n$
$= (k-1)n^{1/k}+16(k-1)$

k-Core

Lemma 1. Every n-node m-edge undirected graph G has a subgraph H so that every node in H has degree $\geq m/n$. H is called *k-core*.

Remark. Such a k-core can be obtained in linear time by iteratively removing the min-degree node. (Why?)

Remark. G has a $\boldsymbol{\delta}$-core.

Let $\boldsymbol{\delta} = e(G)/n$
$= (k-1)n^{1/k}+16(k-1)$

$\boldsymbol{\Theta}_{2k}$-graph

Definition. A $\boldsymbol{\Theta}_{2k}$-graph is a cycle of length $\geq 2k$ with a chord.

Lemma 2. For each $k \geq 3$, every bipartite graph of min degree k has a $\boldsymbol{\Theta}_{2k}$-subgraph. [Hint. the longest path]

Lemma 3. Let A, B be any partition of the node set of a $\boldsymbol{\Theta}_{2k}$-subgraph F into two subsets. For every $\ell \in [1, 2k-1]$, there exists a length-ℓ AB-path, i.e. a path from a to b for some $a \in A$, $b \in B$, unless F is bipartite and A, B is the bipartition of F.

BFS on a $\boldsymbol{\delta}$-core H of G

Let x be any node in H, and set x as the roof of our BFS.

Let D_i be the set of nodes of depth i. For example, $D_0 = \{x\}$ and $D_1 = \{u : \text{edge } (u, x) \text{ in } H\}$.

Let $H[D_i]$ be the induced subgraph of D_i in H, and let $H[D_i, D_{i+1}]$ be the induced bipartite subgraph of D_i and D_{i+1} in H.
($D[D_i, D_{i+1}] \neq H[D_i \cup D_{i+1}]$)

Lemma 4. $H[D_i]$ and $H[D_i, D_{i+1}]$ cannot contain $\boldsymbol{\Theta}_{2k}$ as a subgraph for each $i \in [1, k-1]$. Otherwise, one can construct a C_{2k} in H (also G). $\rightarrow\leftarrow$ (by Lemma 3)

Sparse $H[D_i]$ and $H[D_i, D_{i+1}]$

Lemma 2. For each $k \geq 3$, every bipartite graph of min degree k has a $\boldsymbol{\Theta}_{2k}$-subgraph.
Lemma 4. $H[D_i]$ and $H[D_i, D_{i+1}]$ cannot contain $\boldsymbol{\Theta}_{2k}$ as a subgraph for each $i \in [1, k-1]$.

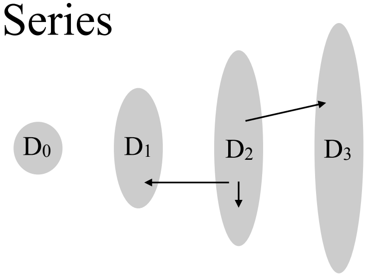
Lemma 5. $\deg(H[D_i]) \leq 4k-4$ and $\deg(H[D_i, D_{i+1}]) \leq 2k-2$ for every $k \geq 2$, every $i \in [1, k-1]$.

Proof. Otherwise $H[D_i, D_{i+1}]$ has a bipartite k-core, yielding a contradiction (Lemma 3 and 4). [req. $k \geq 3$]

Otherwise $H[D_i]$ has a bipartite subgraph I whose $\deg(I) > 2k-2$, implying that $H[D_i]$ has a bipartite k-core. $\rightarrow\leftarrow$ [req. $k \geq 3$]

For k=2, it suffices to check D_0, D_1, D_2 by the C_4-freeness of H.

Power Series



1. The nodes in H (also D_2) have average degree $\geq \boldsymbol{\delta}$.
2. D_2 only have edges to D_1, D_2, and D_3. (Why?)
3. Together with Lemma 5 and an induction, each node in D_2 has $\boldsymbol{\delta}$ - O(k) edges to D_3 but each node in D_3 has O(k) edges to D_2.
4. This assymetry implies that $|D_3|/|D_2|$ is large for large $\boldsymbol{\delta}$.
5. The above holds for D_i, $i \leq k$. Thus the sequence $|D_1|, |D_2|, \ldots, |D_k|$ grows exponentially. If $\boldsymbol{\delta}$ is large, then $|D_k|$ exceeds n. $\rightarrow\leftarrow$