Massive Data Algorithmics

The Streaming Model

Lecture 18: Graph Streams

Graph Streams

- The input streams consists of tokens $(u,v) \in [n] \times [n]$, describing the edges of a simple graph G on vertex set $[n]$.
- We assume each edge of G appears exactly once in the stream.
- The number n is known beforehand but m, the length of the stream and the number of edges in G, is not.
- Both directed and undirected graph can be considered in this model but we will only study undirected graphs; so we may assume that the tokens describe doubleton sets $\{u,v\}$.
- Unfortunately, we mostly need provabley $\Omega(n)$ space in this model, even allowing multipass over the input stream.
- Therefore, our holy grail is to use $O(n\log^c n)$ space.
- Algorithms achieving such a space bound are sometimes called semi-streaming algorithms.

Connectedness Problem

- The input graph G is a graph stream.
- Output is 1 if G is connected and 0 if not. So we need an exact answer.

Algorithm

Initialize : $F \leftarrow \emptyset, X \leftarrow 0$;

Process $\{u, v\}$:

if $\neg X \wedge (F \cup \{\{u, v\}\}$ *does not contain a cycle*$)$ **then**

  $F \leftarrow F \cup \{\{u, v\}\}$;

  if $|F| = n - 1$ **then** $X \leftarrow 1$;

Output : X ;

Intuition

- For this problem, as well as many others, the algorithms will consist of maintaining a subgraph of G satisfying certain conditions.
- For connectedness, the idea is to maintain a spanning forest F of G.
- As G gets updated, F might or might not become a tree at some point. Clearly G is connected iif it does.

Analysis

- The correctness is clear.
- space: $O(n \log n)$ bits
- Union-Find data structure can be used to run the algorithm quickly.
- Note that this algorithm assume an insertion-only graph stream: edges only arrive and never depart from the graph.

Bipartiteness Problem

- The input graph G is a graph stream.
- Output is 1 if G is bipartite and 0 if not. So we need an exact answer.

Algorithm

Initialize : $F \leftarrow \phi$, $X \leftarrow 1$;

Process $\{u, v\}$:

```
if X then
    if F ∪ {{u, v}} does not contain a cycle then
        F ← F ∪ {{u, v}} ;
    else if F ∪ {{u, v}} contains an odd cycle then
        X ← 0 ;
```

Output : X ;

Intiution

- A graph G is bipartite iff its vertices can be colored using 2 colors, or equivalently it does not have an odd cycle.
- Being bipartite is a monotone property, i.e. given a non-bipartite graph, adding edges to it can not make it bipartite.
- Therefore, once a streaming algorithm detect that the edges seen so far make the graph non-bipartite, it can stop doing more work.

Analysis

- space: $O(n \log n)$ bits
- Suppose the algorithm output 0. Then G must contain an odd cycle. This cycle does not have a 2-coloring, so neither G.
- Now, suppose the algorithm output 1. Let $\chi : [n] \to \{0,1\}$ be a 2-coloring of F. We claim that χ is a 2-coloring for G
  - Consider an edge $e = \{u,v\}$ of G.
  - If $e \in F$, we already know that $\chi(u) \neq \chi(v)$.
  - Otherwise, $F \cup \{e\}$ must contain an even cycle.
  - Let π be the path in F obtained by deleting e from this cycle. Then π runs between u and v and has odd length.
  - Since every edge on π is colored by χ, we again get $\chi(u) \neq \chi(v)$.

Spanners

- $d_G(u,v)$ is defined to be the length of the shortest path from u to v in G.
- The input is a graph stream G and an integer t
- For a query pair (u,v), output a t-approximation of $d_G(u,v)$.

Algorithm

Initialize : $H \leftarrow \varnothing$;

Process $\{u, v\}$:

if $d_H(u, v) \geq t + 1$ **then**

  $H \leftarrow H \cup \{\{u, v\}\}$;

Output : On query (x, y), report $\hat{d}(x, y) = d_H(x, y)$;

Intiution

- The algorithm maintains a subgraph H of G with the property that $\forall u, v : d_G(u,v) \leq t \cdot d_H(u,v)$.
- Indeed, H approximates distances in G with a factor of t.
- Such a subgraph of G is called a t-spanner of G.

Analysis

- Pick any two vertices u and v.
- If $d_G(u,v) = \infty$, then clearly $d_H(u,v) = \infty$ as well, and we are done.
- Otherwise, let $\pi = v_0, \cdots, v_k$ be the shortest path from $v_0 = u$ to $v_k = v$ in G. We have $d_G(u,v) = k$.
- By the triangle inequality: $d_H(u,v) \leq \sum_{i=0}^{k-1} d_H(v_i, v_{i+1})$
- If $e = \{v_i, v_{i+1}\}$ exists in H, then $d_H(v_i, v_{i+1}) = 1$.
- Otherwise $e \notin H$ which means that at the time e appeared in the input stream, we had $d_{H'}(v_i, v_{i+1}) \leq t$, where H' was the value of H at that time. Since H' is a subgraph of H, we have $d_H(v_i, v_{i+1}) \leq t$ as well.
- Thus, $d_H(u,v) \leq \sum_{i=0}^{k-1} d_H(v_i, v_{i+1}) \leq t \cdot k = t \cdot d_G(u,v)$

The Size of a Spanner: High-Girth Graphs

- The girth $\gamma(G)$ of a graph G is defined to be the length of its shortest cycle; we set $\gamma(G) = \infty$ if G is acyclic.
- The graph H constructed by the algorithm has $\gamma(H) \geq t+2$.
- The following theorem places an upper bound on the size of a graph with high girth.

Theorem. Let n be sufficiently large. Suppose the graph G has n vertices, m edges, and $\gamma(G) \geq k$ for an integer k. Then

$$m \leq n + n^{1+\frac{1}{\lfloor (k-1)/2 \rfloor}}$$

The Size of a Spanner: High-Girth Graphs

- Let $d = 2m/n$ be the average degree of G.
- If $d \leq 3$, then $m \leq 3n/2$ and we are done.
- Otherwise, let F be the subgraph of G obtained by repeatedly deleting from G all vertices of degree less than $d/2$.
- F has the minimum degree at least $d/2$ and is nonempty, because total number of edges deleted is less than $n \cdot d/2 = m$.
- Put $\ell = \lfloor \frac{k-1}{2} \rfloor$. Clearly, $\gamma(F) \geq \gamma(G) \geq k$.
- For any vertex v of F, the ball in F centered at v and of radius ℓ is a tree (otherwise, it contains a cycle of length $2\ell \leq k-1$).
- By the minimum degree property of F, when we root this tree at v, its branching factor is at least $d/2 - 1 \geq 1$. Therefore the tree has at least $(d/2-1)^\ell$ vertices.
- It follows that $n \geq (\frac{d}{2} - 1)^\ell = (\frac{m}{n} - 1)^\ell$ which implies $m \leq n + n^{1+\frac{1}{\ell}}$

The Size of a Spanner: High-Girth Graphs

- Using $\lfloor \frac{k-1}{2} \rfloor \geq \frac{k-2}{2}$, we can weaken the bound to $m = O(n^{1+\frac{2}{k-2}})$
- Plugging in $k = t + 2$, we see that the t-spanner H constructed by the algorithm has $|H| = O(n^{1+\frac{2}{t}})$.
- Thereofore, the space used by the algorithm is $O(n^{1+\frac{2}{t}} \log n)$ bits.
- In particular, we can 3-approximate all distances in a graph by a streaming algorithm in space $\tilde{O}(n^{5/3})$

References

- **Data Stream Algorithms** (Chapter 13)
  Lecture notes by A. Chakrabbarti and D. College