**CS 594: Representations in Algorithm Design** **Spring 2022**

# Lecture 4: 01/20/2022

*Lecturer: Xiaorui Sun* *Scribe: Parth Deshpande*

## 1 Today and next Lecture

- Today: Trees for distance (Low Stretch Spanning Tree)
- Next Lecture: Extenders

## 2 Low Stretch Spanning Tree

### 2.1 Graph Distance(d(x, y))

- A graph can have weighted or unweighted edges. For example, in an airline network, the timie taken to reach from one city to another is equivalent to the graph distance.
- Here, d(x, y) = distance between vertices x, y = sum of edge weights for all edges in the path which is the minimum of all paths connecting x - y.
- Dijkstras algorithm is used to compute shortest path from x - y if graph has non-negative edge weights.

Question: For an unweighted graph, can you compute d(x, y) in time proportional to O(d(x, y))? (The runtime should be proportional to length of path)

Intuitive idea: BFS may compute. However in BFS, we have to check all vertices for each level. Hence it would be linear in number of vertices (n).

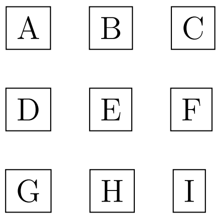
Now, consider a rooted tree structure in which we know the parent for each vertex. In this structure, we alternatively find path x to root and y to root until we get a common ancestor for both the vertices.

Extending this idea to a general graph, process the general graph and produce a tree such that the distance within the tree is close to the general graph.

Goal: Find the spanning tree of graph G such that for most vertex pairs (x, y), the distance.

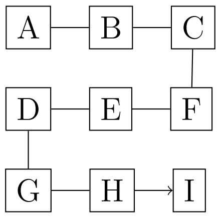
$$D_T(x,y) \approx D_G(x,y)$$

Diameter: $max(D(x,y))$
Consider a grid graph with 'n' total vertices. (Here, Diameter = $\sqrt{2n}$)



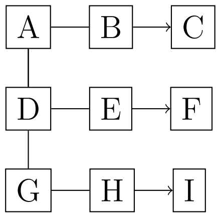
Consider the following trees:

1.



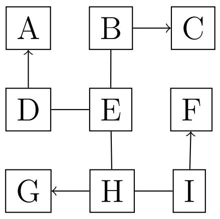
Here, Diameter = $O(n)$ Hence, this is not a good tree.

2.



Here, Diameter = $O(\sqrt{n})$ However, consider distance $D(C,F)$:
$OriginalDistance = 1$
$Distanceintree = 2\sqrt{n} + 1$

Correct Approach: The best solution is given by a random shortest path tree from a random vertex.



### 2.1.1 Stretch of a tree

Consider a graph G and a spanning tree T.

If G is unweighted,

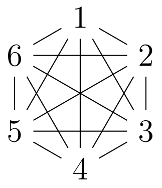
$$stretch(u,v) = D_T(u,v)$$

If G is weighted,

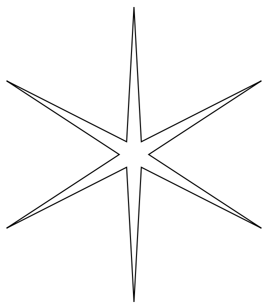
$$stretch(u,v) = \frac{D_T(u,v)}{W(u,v)}$$

Examples:

1. Complete Graph (All vertices have edges between them)

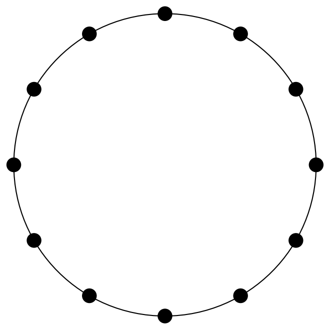


Consider the following diagram (Assume nodes at the end of the arms of the star and one node at the center of the star(x)):

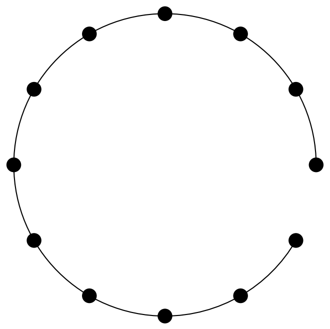


Here, $stretch(x, othernode) = 1$ and $stretch(othernode, othernode) = 2$

2. Circle Graph



Consider the following diagram:



Here, $stretch(best) = 1$ and $stretch(worst) = n - 1$ Hence, not all edges have a good stretch.

#### 2.1.2 Total Stretch

$$Stretch_T(G) = \sum_{e \epsilon E} Stretch_T(e)$$

Goal: Construct spanning tree with small stretch (ideally $O(m)$)(m = Number of edges)

Practically, currently obtained best $stretch = O(mlog(n)log(log(n)))$

Theoretically, there exists a graph with $stretch >= \Omega(mlog(n))$