Succinct Graph Structures and their Applications	**Spring 2018**

Lecture 7: May 24

Lecturer: Merav Parter

Low-Stretch Subgraph Trees

In the previous class, we presented an embedding into a distribution of trees with expected stretch $O(\log n \cdot \log \mathtt{Diam}(G))$. These trees were not subgraphs of G and contained vertices not in G. The goal of this class is to construct a distribution over trees which are subgraphs of G. We will show the following theorem due to Elkin, Emek, Spielman and Teng [EEST08].

Theorem 7.1 ([EEST08]) *Any n-vertex graph can be embedded into a distribution of spanning trees (subgraphs of G) with distortion of $\alpha = O(\log^3 n)$.*

The state-of-the-art for this problem is $\alpha = O(\log n \cdot \mathtt{poly}(\log\log n))$ by [ABN08]. In this class, we will see a simplified version of the EEST algorithm by Dhamdhere, Gupta and Räcke [DGR06].

Recap of Last Week's Tree Embedding. The computation of the tree distribution from last week was based on the low-diameter decomposition (LDD). The algorithm decomposes G into into vertex-disjoint components $G_1, \ldots, G_k$ that have small weak diameter with respect to G. These components got connected in the virtual tree by adding a dummy root vertex (that represent the graph G in the tree). Insisting on having a subgraph tree of G gives raise of the following complication. The LDD procedure breaks G into components that might be connected in an arbitrarily bad manner in G. That is, we are no longer allowed to add a dummy vertex to connect these components and rather forced to use true G-edges. This might be problematic when the diameter of the contracted graph (obtained by contracting each component G_i into a single super-node) is $\Omega(\mathtt{Diam}(G))$. To add more insult to this injury, this $\Omega(\mathtt{Diam}(G))$ bound is only for one recursion level. Applying the procedure recursively in each component G_i might end up with a tree with an expected stretch of $\Omega(n)$. To overcome this technicality, we need a more restrictive LDD procedure which also guarantees that the output components can be connected (in the contracted graph view) by a constant-depth tree. For that purpose, EEST introduces the notion of star-decomposition.

Definition 7.2 (Star-Decomposition) *A star-decomposition of a graph G with a designated root node r_0 is a set of vertex disjoint connected components $G_0 = (V_0, E_0), \ldots, G_k = (V_k, E_k)$ together with a collection of root nodes $r_0, \ldots, r_k$ where $r_i \in V_i$ such that each r_i has a neighbor in V_0. In a δ-star-decomposition, each component has a radius of $\mathtt{Rad}(r_i, G_i) = \max_{u \in G_i} \mathtt{dist}_{G_i}(r_i, u) \leq \delta$.*

We first describe the construction of the tree given an algorithm that computes the star-decomposition. For a given root vertex r_0, let $\Delta = \mathtt{Rad}(r_0, G)$.

Algorithm $\mathsf{TESubGraph}(G, r_0, \Delta)$

1. Compute a $(7/8 \cdot \Delta)$-Star-Decomposition resulting in components $G_0, \ldots, G_k$ with roots $r_0, \ldots, r_k$.
2. For each G_i, $T_i \leftarrow \mathsf{TESubGraph}(G_i, r_i, 7/8 \cdot \Delta)$.
3. Let $T = \bigcup_i T_i \cup \{(r_i, q_i)\}$ where q_i is a neighbor of r_i in G_0.

Figure 7.1: Algorithm for computing a low-stretch tree $T \subseteq G$ using star-decomposition

Computing the Star-Decomposition

Algorithm StarDecomp has two phases. The first constructs the main component G_0 containing the root node r_0. The second procedure constructs the components $G_{i\geq 1}$ that will be connected to G_0.

(A) Construct G_0 via forward cut.

1. Choose a radius γ uniformly in $[\Delta/4, \Delta/2]$.
2. Cut all edges (u, v) at distance γ from r_0, i.e., $\texttt{dist}_G(r_0, u) \leq \gamma < \texttt{dist}_G(r_0, v)$.
3. The output subgraph G_0 is the connected component containing r_0.

(B) Construct G_i's via backward cut. Let $x_1, \ldots, x_s$ denote the vertices with at least one neighbor in V_0. We call these vertices *portal nodes*. The subgraphs $G_1, \ldots, G_k$ are computed by cutting pieces from $G \setminus V_0$ using a random radius selection. Unlike the previous construction, here the balls around portal nodes are computed based on a *backward-edge distance*. As we are going to see, this distance definition guarantees that each component G_i contains at least one portal node x_i, and hence all components $G_{i\neq 0}$ are directly connected to G_0 (hence, it is called star-decomposition). The backward distance is defined as follows. Each edge $\{u, v\}$ in $G \setminus V_0$ is replaced by two directed edges (u, v) and (v, u), where:

$$\ell(u,v) = \begin{cases} 1, \texttt{dist}(r_0, v) = \texttt{dist}(r_0, u) - 1 \\ 1, \texttt{dist}(r_0, v) = \texttt{dist}(r_0, u) \\ 0, \texttt{dist}(r_0, v) = \texttt{dist}(r_0, u) + 1 \end{cases}$$

This function $\ell(\cdot, \cdot)$ is defined so that an edge has a nonzero length iff either the edge is not in any BFS tree of G rooted at r_0, or otherwise it is directed towards r_0 in such a tree. The length of an x-y shortest path counts how many times the distance to r_0 does not increase while going from x to y.

All vertices are initially unmarked. As long as there exists an unmarked *portal* node x_i, a new component G_i is constructed in the following manner:

- Start a region growing by picking a radius γ' from the Geometric distribution $\mathsf{Geom}(p)$ for $p = 32\log^2 n/\Delta$ from x_i.
- The component G_i contains all the unmarked vertices at backward-distance at most γ' from x_i.

The key motivation for defining this backward distance is the following property.

Observation 7.3 *Whenever there is an unmarked node, there is an unmarked portal node.*

Proof Sketch: Let T_{r_0} be the BFS tree rooted at r_0 in G. For every portal node x_i, let T_{x_i} be the subtree of T_{r_0} rooted at x_i. Since the backward distance from r_0 to each vertex in T_{x_i} is *zero*, we get that when x_i is assigned to a cluster, its entire subtree is assigned to the same cluster. That is, the edges that are cut by the backward cut procedure are necessarily edges with a nonzero backward length, hence the BFS edges are never cut.

The correctness of the algorithm is based on the next three lemmas.

Lemma 7.4 *W.h.p., the radius of each G_i is at most $7/8\Delta$.*

Lemma 7.5 *The probability that an edge (u, v) is cut when called to* StarDecomp(G, r_0, Δ) *is $O(\log^2 n/\Delta)$.*

Lemma 7.6 *If an edge (u, v) is cut in level i (of the recursion) when called to* StarDecomp(G', r'_0, Δ_i)*, then $\texttt{dist}_T(u, v) = O(\Delta_i)$.*

We first complete the stretch argument on the output tree T (of Alg. TESubGraph) assuming that the above lemmas hold. By Lemma 7.4, there are $O(\log n)$ recursive levels. Fix an edge (u, v) and let A_i be the event that (u, v) is cut in level i. We have that:

$$\begin{aligned} \mathbb{E}(\mathtt{dist}_T(u,v)) &= \sum_{i=1}^{O(\log n)} \Pr\big[A_i \ \mid \ \bar{A}_{i-1} \wedge \ldots \wedge \bar{A}_1\big] \cdot O(\Delta_i) \\ &= \sum_{i=1}^{O(\log n)} O(\log^2 n/\Delta_i) \cdot O(\Delta_i) = O(\log^3 n) \ . \end{aligned}$$

We use the next auxiliary claim that follows immediately by the properties of the geometric distribution.

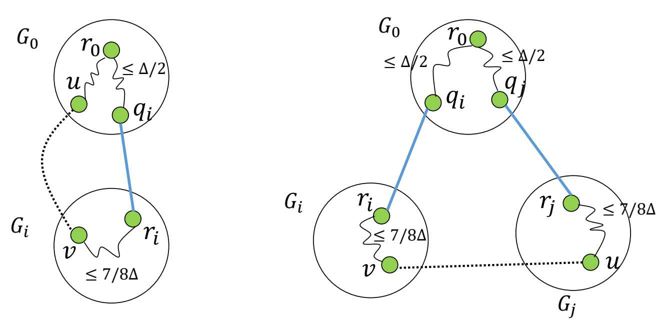
Claim 7.7 *The backward radius* γ' *is: (1) at most* $\Delta/16$ *with probability* $1 - 1/n^{O(\log n)}$ *and (2) at most* $\Delta/(16 \log n)$ *with probability* $1 - 1/\mathtt{poly}(n)$.

Proof of Lemma 7.4: Consider a vertex v in G_i centered at the portal vertex r_i. Let P be the r_i-v shortest-path in G_i with respect to the backward distance. Since $v \in G_i$, we have that $\ell(r_i, v) \leq \gamma'$ and thus there are at most $\Delta/16$ backward edges on P. By Cl. 7.7(1), this holds for all subgraphs in all recursion levels. This implies that there is a path between v and r_0 that have at least $\Delta/4 + |P| - 2\Delta/16$ forward-edges, sine $\mathtt{dist}_G(r_0, v) \leq \Delta$, we have that $\Delta/4 + |P| - 2\Delta/16 \leq \Delta$. Hence, $|P| \leq 7/8\Delta$ are required.

Proof of Lemma 7.4: Consider a fixed edge (u, v), where without loss of generality $\mathtt{dist}_G(r_0, u) \leq \mathtt{dist}_G(r_0, v)$. The edge (u, v) is cut in phase (A) if $\mathtt{dist}_G(r_0, v) = \mathtt{dist}_G(r_0, u) + 1$ and $\gamma \in [\mathtt{dist}_G(r_0, u), \mathtt{dist}_G(r_0, v))$. Since γ is chosen from a uniform distribution of width $\Delta/4$, we get that (u, v) is cut with probability $4/\Delta$.

We next bound the probability of an edge (u, v) to be cut during the backward cut phase (B). Without loss of generality, assume that u becomes clustered by joining V_i, rooted at r_i, not before v. Let γ' be the backward radius sampled by r_i from the geometric distribution. Observe that (u, v) can be cut only if the directed edge $u \rightarrow v$ has nonzero length in the backward distance. The probability that (u, v) is cut in such a case if $\Pr[\gamma' < \ell(r_i, v) \ \mid \ \gamma' \geq \ell(r_i, u)] = p = O(\log^2 n/\Delta)$. The lemma follows.

Proof of Lemma 7.6: For the purpose of proving this lemma, it is convenient to view the algorithm as an iterative one. In each iteration i, we have a subgraph $\widehat{G}_i$ consisting of the subset of edges already added to the tree, and a collection of graph components. Initially, $\widehat{G}_1 = G$ and at the end of the algorithm, $\widehat{G}_k = T$ for $k = O(\log n)$.



Consider an edge (u, v) that got cut in the first level of the recursion where G is decomposed into $G_0, \ldots, G_k$. The subgraph $\widehat{G}_2$ contains the union of all G_i's graphs and the star edges (q_i, r_i) connecting the root of each G_i to G_0. If (u, v) is cut in either the forward phase or the backward phase, there is an u-v path of length at most 3Δ in $\widehat{G}_2$, see the figure above for an illustration. However, this holds immediately after the decomposition, and throughout the recursion levels the distance from q_i to its root r_i might increase. We next show that this node to root distance is increased by a factor of at most $(1+1/\log n)$ in each iteration, and as there are $O(\log n)$ iterations, the stretch is increased by at most $(1+1/\log n)^{O(\log n)} = O(1)$ factor.

Claim 7.8 *In each iteration i, the distance from node to root is increased by factor $(1+1/\log n)$.*

Proof: Each x-y path in iteration i is a concatenation of node to root paths (see the below figure). We will analyze how the path length from root to node is increased in iteration i. Consider the component G_{i_j} (see figure) and let Δ_j be the radius of G_{i_j} with respect to r_{i_j}. In iteration i, a star-decomposition is computed on each G_{i_j} subgraph. We now bound the distance between v_j and r_{i_j} due to this decomposition, i.e., in the graph $\widehat{G}_{i+1}$. First, if $\mathtt{dist}_{G_{i_j}}(r_{i_j}, v_{i_j}) \leq \Delta_j/4$, then v_{i_j} is in the component of r_{i_j} when applying Alg. StarDecomp on G_{i_j}. More generally, if r_{i_j} and v_{i_j} are not separated in the decomposition, their in $\widehat{G}_{i+1}$ is unchanged. Consider now the case that r_{i_j} and v_{i_j} are separated, and let r' be the root component of $v_{i,j}$. Let P be the shortest-path between r' and v_{i_j} based on the backward distance. This path has at most $\Delta/(16\log n)$ backward edges based on Cl. 7.7(2). We therefore have that

$$\mathtt{dist}_{\widehat{G}_{i+1}}(r_{i_j}, v_{i_j}) \leq \mathtt{dist}_{\widehat{G}_i}(r_{i_j}, r') + |P| \; . \tag{7.1}$$

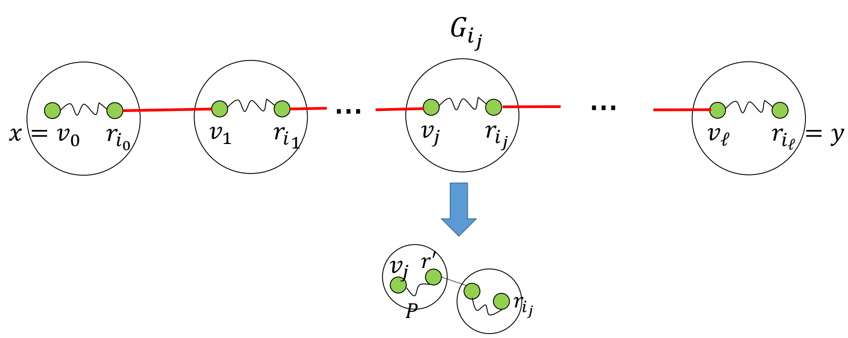
By using a similar argument to before, there is an $r_{i_j} - v_{i_j}$ path with at least $\mathtt{dist}_{\widehat{G}_i}(r_{i_j}, r') + |P| - \Delta/(8\log n) \leq \mathtt{dist}_{\widehat{G}_i}(r_{i_j}, v_{i_j})$ forward edges. Thus,

$$|P| \leq \mathtt{dist}_{\widehat{G}_i}(r_{i_j}, v_{i_j}) - \mathtt{dist}_{\widehat{G}_i}(r_{i_j}, r') + \Delta_j/(8\log n).$$

By plugging this into Eq. (7.1), we get that:

$$\begin{aligned}\mathtt{dist}_{\widehat{G}_{i+1}}(r_{i_j}, v_{i_j}) &\leq \mathtt{dist}_{\widehat{G}_i}(r_{i_j}, v_{i_j}) + \Delta_j/(8\log n) \\ &\leq (1+1/\log n) \cdot \mathtt{dist}_{\widehat{G}_i}(r_{i_j}, v_{i_j}) \; ,\end{aligned}$$

where the last inequality follows by the fact that $\mathtt{dist}_{\widehat{G}_i}(r_{i_j}, v_{i_j}) \geq \Delta_j/4$. ■



References

[ABN08] Ittai Abraham, Yair Bartal, and Ofer Neiman. Nearly tight low stretch spanning trees. In *Foundations of Computer Science, 2008. FOCS'08. IEEE 49th Annual IEEE Symposium on*, pages 781–790. IEEE, 2008.

[DGR06] Kedar Dhamdhere, Anupam Gupta, and Harald Räcke. Improved embeddings of graph metrics into random trees. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 61–69. Society for Industrial and Applied Mathematics, 2006.

[EEST08] Michael Elkin, Yuval Emek, Daniel A Spielman, and Shang-Hua Teng. Lower-stretch spanning trees. *SIAM Journal on Computing*, 38(2):608–628, 2008.