Advanced Distributed Algorithms **Spring 2019**

Lecture 2: April 4

Lecturer: Merav Parter

Network Decomposition – Part II

We continue with network decomposition and focus on (d, c) decompositions for $d, c = O(\log n)$. We first describe a sequential deterministic construction, and then show how to get an efficient distributed construction when introducing randomness.

Sequential Alg. for Network Decomposition. The algorithm has $O(\log n)$ iterations, in each iteration i, it outputs a collection of clusters $\mathcal{C}_i$. These clusters are vertex disjoint, non-neighboring, and have diameter of $O(\log n)$. Since all the clusters in $\mathcal{C}_i$ are not neighbors[1], we can color all these clusters with the same color i. Since there are $O(\log n)$ iterations, we will use $O(\log n)$ colors over all.

We now zoom into the first iteration, and later on explain the general case. We will compute the clusters one by one as follows. Pick a vertex u and grow a ball around it until reaching to the smallest value r satisfying that:

$$|B_G(u, r+1)| \leq 2|B_G(u, r)| \ . \tag{2.1}$$

Add the ball $B_G(u, r)$ to the list of clusters, and omit the ball $B_G(u, r+1)$ from G. We repeat this procedure, and pick the next vertex u', grow a ball around it, until the entire graph G is empty. This completes the description of a single iteration. Note that we do not require the graph G to be connected, and in fact throughout the iterations, the graph induced on the unclustered graph might indeed be disconnected. This thus not affect the behavior of the algorithm. By Eq. (2.1), we immediately get:

Observation 2.1 *(1) The diameter of all clusters is $O(\log n)$, and (2) the number of clustered vertices (in this iteration) is at least as large as the number of unclustered vertices.*

Property (2) of Observation 2.1 implies that after $O(\log n)$ iterations, all vertices are clustered. It remains to see that indeed all clusters C, C' generated in a given iteration do not have a neighboring pair, i.e., that there exists no $u \in C$ and $v \in C'$ such that $(u, v) \in E$. This follows by our construction: whenever we add the cluster $B_G(u, r)$, we omit all the neighbors of this clusters from G, by removing the larger ball $B_G(u, r+1)$. The code for the i^{th} iteration is in Fig. 2.2. See Fig. 2.1 for an illustration of the clusters created in the first iteration.

Randomized Dist. Alg. for (Weak) Network Decomposition. The above mentioned algorithm provides us network decomposition with essentially the best possible bounds (why? see here [LS93] if you are curios). The only drawback of this algorithm is that it is highly non-distributed: clusters are computed one after the other. We next see a distributed construction which follows the same high-level idea but uses randomness to allow the parallelism of the ball growing procedure.

Lemma 2.2 *[LS93] For every graph $G = (V, E)$, there is a randomized distributed algorithm that computes an $(O(\log n), O(\log n))$ weak network decomposition using $O(\log^2 n)$ rounds, w.h.p.*

The construction has been "fixed" by Elkin and Neiman [EN16] to provide a strong decomposition with the same bounds on c, d and round complexity.

[1] In the cluster graph

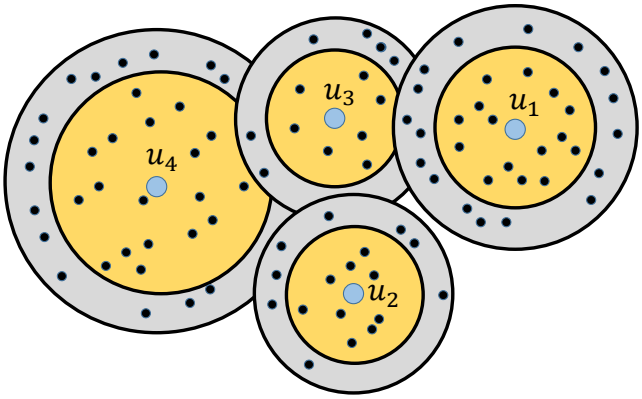

Figure 2.1: The labels of the vertices describe the traversal order of the algorithm. The first vertex picked is u_1, the algorithm grows a ball around it as long as some expansion criteria is satisfied. The shell of the ball (shown in gray) is omitted from the graph. This is important in order to make sure that the output clusters are not neighbors in the cluster graph, and thus can be given the same color. All nodes that fall in the gray regions will be handled in future iterations.

The i^{th} Iteration of Algorithm SeqBallGrowing(G)**:**

Input: Graph G_i induced on all vertices that are not yet clustered.
Output: Collection of clusters $\mathcal{C}_i = \{C_1, \ldots, C_\ell\}$ of vertices in G_i, such that all clusters are vertex disjoint, with depth $O(\log n)$, and are non-neighboring in G_i.

- Initialize $\mathcal{C}_i \leftarrow \emptyset$.
- While G_i is not empty do:
  - Pick an arbitrary node u in G_i.
  - Grow a ball around u in G_i up to the smallest value r satisfying:

$$|B_{G_i}(u, r+1)| \leq 2|B_{G_i}(u, r)| \ .$$

  - Add $B_{G_i}(u, r)$ to $\mathcal{C}_i$ and omit $B_{G_i}(u, r+1)$ from G_i
- Color all clusters in $\mathcal{C}_i$ with color i.

Figure 2.2: Sequential Construction of $(O(\log n), O(\log n))$ Network Decomposition via Ball Growing

Just like in the sequential algorithm, this algorithm too has $O(\log n)$ phases. In each phase i, it computes a collection $\mathcal{C}_i$ of non-neighboring vertex disjoint clusters of weak diameter $O(\log n)$. All the clusters in $\mathcal{C}_i$ will be given a color i.

We now describe the i^{th} phase, and let G_i be the current graph of the yet unclustered vertices (initially, $G_1 = G$). Instead of growing the balls one by one, we lets them all grow together as follows. Every vertex u

samples a radius r_u from the geometric distribution[2] with parameter $p = 0.5$. That is, for every y:

$$Pr[r_u = y] = p \cdot (1-p)^{y-1}.$$

The vertex u then sends a message to all nodes in its r_u-ball in G_i. Since we are in the LOCAL model this can be done in $O(r_u)$ rounds. The message sent by u contains the ID of u and the value of its radius. Now, every vertex v considers all the messages that it got, and defines $Center(v) = \{\min ID(u) \mid r_u \geq \texttt{dist}(u,v)\}$. That is, v selects as its center the vertex of minimum ID out of all vertices from which it has received messages. Finally, v becomes unclustered if $\texttt{dist}(Center(v), v) = r_{Center(v)}$. This defines clusters potentially around vertex u. The phase ends by eliminating all clustered vertices from G_i.

Observation 2.3 (Small diameter) *W.h.p. $r_u = O(\log n)$ for every u and thus the weak diameter of all clusters is $O(\log n)$.*

Note that since we omit the last layer from every cluster, the clusters computed in iteration i are non-neighbors and can be colored with color i. The most interesting part is to show that after $O(\log n)$ iterations all vertices are clustered, and thus $O(\log n)$ colors are sufficient. Towards that goal, we show:

Lemma 2.4 *$Pr[v$ is unclustered$] \leq p$*

Proof: For simplicity, we consider the probability that v becomes unclustered in the first iteration. Same analysis applies for any other iteration. We have:

$$Pr[v \text{ is unclustered}] = \sum_{u \in V} Pr[v \text{ is unclustered} \mid Center(v) = u] \cdot Pr[Center(v) = u] \ .$$

We will now zoom into a fixed node u and bound the probability $Pr[v \text{ is unclustered} \mid Center(v) = u]$. Define three events:

- D_u: $r_u = \texttt{dist}(u, v, G)$,
- E_u: $r_u \geq \texttt{dist}(u, v, G)$, and
- F_u: for all $u' < u$, $r_{u'} < \texttt{dist}(u', v, G)$.

We then have the following:

$$\begin{aligned} Pr[v \text{ is unclustered} \mid Center(v) = u] &= \frac{Pr[v \text{ is unclustered and } Center(v) = u]}{Pr[Center(v) = u]} \\ &= \frac{Pr[D_u \text{ and } F_u]}{Pr[E_u \text{ and } F_u]} = Pr[D_u]/Pr[E_u] = p \ , \end{aligned}$$

where the penultimate inequality is due to the fact that F_u is independent of the events D_u and F_u, the last inequality is by the definition of the geometric variable. Overall, we get that $Pr[v \text{ is unclustered}] = p \cdot \sum_{u \in V} Pr[Center(v) = u] = p$. ■

By Lemma 2.4, we get that the probability that a vertex is unclustered for $c \log n$ phases is at most $p^{c \log n} \leq 1/n^c$. Therefore, we have at most $O(\log n)$ phases, each phase is implemented in $\max_u r_u = O(\log n)$ rounds w.h.p., thus the total round complexity is $O(\log^2 n)$.

[2]Imagine the node flips a coin that comes up heads with probability p. The radius of the vertex is the first time that the coin flip came out heads.

References

[EN16] Michael Elkin and Ofer Neiman. Distributed strong diameter network decomposition. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, pages 211–216. ACM, 2016.

[LS93] Nathan Linial and Michael Saks. Low diameter graph decompositions. *Combinatorica*, 13(4):441–454, 1993.