ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

FS 2012

Prof. R. Wattenhofer
Jara Uitto

Principles of Distributed Computing
Exercise 10

1 Distributed Network Partitioning

In this exercise, we will derive an asynchronous distributed version of the cluster construction algorithm presented in the lecture.

Assume that (in $O(n)$ time and using $O(m + n \log n)$ messages) a spanning tree has already been computed. We further assume that the constant ρ in the algorithm is 2. Moreover, you can ignore the intercluster edges in this exercise, i.e. the number of intercluster edges does not have to be reduced once the clusters are built.

Just like the centralized algorithm, the distributed algorithm repeatedly applies the following two steps to construct a partitioning:

1. Find a (cluster) leader.
2. Construct a cluster C, remove all the nodes $v \in C$ and remove all the edges $\{u, v\}$ for which $u \in C$ or $v \in C$ or both.

We shall now develop how this algorithm can be executed in an asynchronous setting.

a) We need a first leader to start the algorithm. Describe how a leader can be determined on the spanning tree in $O(n)$ time using $O(n)$ messages!

Given a leader, we need to build the cluster by adding more and more nodes to the cluster.

b) Describe how the leader constructs the cluster! Let C denote the constructed cluster. Show that the time complexity to construct the cluster is bounded by $O(|C|)$!

Hint: If the radius of C is r, show that the time complexity is $O(r^2) \subseteq O(\log^2 |C|) \subset O(|C|)$!

c) Let $E' \subseteq E$ be the set of edges that can be removed in Step 2. Show that the construction of cluster C can be accomplished using $O(|E'|)$ messages!

Hint: Use the observation that edges closer to the leader have to be traversed more often, but there are more edges with a greater distance to the leader!

d) Once a cluster has been constructed, we need to find the next leader in the remaining graph. Show how this task can be done in $O(n)$ time using $O(n)$ messages for all rounds in total.

Hint: Think about how all nodes in the graph can be visited in $O(n)$ time using $O(n)$ messages in total using the spanning tree! How can this tree-traversal scheme be used to find the cluster leaders every time a new cluster has been built?

e) Putting everything together, show that the entire partitioning requires $O(n)$ time and uses $O(m)$ messages![1]

[1] The construction phase of the spanning tree is (clearly) not considered in these bounds.