CS41 Lab 7

October 21, 2019

The lab problems this week focus on divide and conquer algorithms. The purpose of this lab is to gain practice using the divide and conquer approach to solving problems.

1. **Divide and conquer minimum spanning trees?**

   Lila has a really cool idea for a divide and conquer algorithm which will find a MST. Given a connected, undirected graph $G = (V, E)$ with weighted edges, Lila's algorithm does the following:

   - Divides the graph into two pieces, $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. ($V_1 \cup V_2 = V$ and V_1 and V_2 are disjoint. E_1 is the edges in E with both endpoints in V_1, and E_2 is the edges in E with both endpoints in V_2.)
   - Recursively finds the MSTs M_1 for G_1 and M_2 for G_2.
   - Finds the lowest-weight edge $e = (u, v)$ with $u \in V_1$ and $v \in V_2$.
   - Returns the minimum spanning tree $M_1 \cup M_2 \cup \{e\}$.

   Unfortunately, this algorithm does not work. Give an example input graph G with weights and describe a run of this algorithm where the algorithm does not return a minimum spanning tree on G.

2. You are interested in analyzing some hard-to-obtain data from two separate databases. Each database contains n numerical values (so there are $2n$ values total). You'd like to determine the *median* of this set of $2n$ values, defined as the n-th smallest value.

   The only way you can access these values is through *queries* to the databases. In a single query, you can specify a value k to one of the two databases, and the chosen database will return the k-th smallest value it contains. Since queries are expensive, you would like to compute the median using as few queries as possible.

   - Design an algorithm that finds the median value using at most $O(\log n)$ queries. Full pseudocode is not necessary, but you must clearly explain how it works, and you must handle all edge cases; e.g., do not assume that n is even.
   - Show that your algorithm correctly returns the median.
   - Prove that your algorithm uses only $O(\log n)$ queries.

3. Solve the following recurrence relation (i) using partial substitution, and (ii) using recursion trees.

$$T(n) = 4T(n/3) + 5n \ , \ \text{for all } n > 3$$
$$T(3) = 5$$