AN O(|E|log log|V|) ALGORITHM FOR FINDING MINIMUM SPANNING TREES *

Andrew Chi-chih YAO
Department of Computer Science, University of Illinois, Urbana, Illinois 61801, USA

Received 30 December 1975, revised version received 9 June 1975

Minimum spanning tree, linear median finding algorithm

1. Introduction

Given a connected, undirected graph $G = (V, E)$ and a function c which assigns a cost $c(e)$ to every edge $e \in E$, it is desired to find a spanning tree T for G such that $\Sigma_{e \in T} c(e)$ is minimal. In this note we describe an algorithm which finds a minimum spanning tree (MST) in O(|E|log log|V|) time. Previously the best MST algorithms known have running time O(|E| X log|V|) for sparse graphs [1], and more recently Tarjan [2] has an algorithm that requires O(|E| X $\sqrt{\log|V|}$) time.

Our algorithm is a modification of an algorithm by Sollin [3]. His method works by successively enlarging components of the MST. In the first stage the minimum-cost edge incident upon each node of G is found. These edges are part of the MST sought. The groups of vertices that are connected by these edges are then identified. By shrinking each such group of vertices to a single node, we obtain a new graph with at most $\frac{1}{2}|V|$ nodes. This process is repeated for a number of times, at each stage for a new graph, until finally a single contracted node remains. Clearly each stage of this procedure involves O(|E|) operations, and log|V| stages are necessary in the worst case. Thus this algorithm requires a total of O(|E|log|V|) operations.

In our algorithm, we first partition the set of edges incident with each node v into k levels $E_v^{(1)}, E_v^{(2)}, \ldots, E_v^{(k)}$ so that $c(e) \leqslant c(e')$ if $e \in E_v^{(i)}$, $e' \in E_v^{(j)}$ and $i < j$. This can be done in O(|E|log k) time by repeatedly applying the linear median-finding algorithm [4]. Having accomplished this, we follow basically Sollin's algorithm as outlined above. Note that the number of operations needed in this phase is now reduced to

$$O\left(\frac{|E|}{k}\log|V|\right)$$

since only approximately $|E|/k$ edges have to be examined at each stage to find the minimum-cost edges incident with all the nodes. Therefore, the total number of operations required by our algorithm is

$$O\left(|E|\log k + \frac{|E|}{k}\log|V|\right),$$

which is O(|E|log log|V|) if we choose k to be log|V|.

2. Algorithm

For the moment, assume $|E| \geqslant |V|\log|V|$. If $|E| < |V|\log|V|$, the algorithm needs a slight modification as will be discussed later.

The algorithm uses three sets T, VS and ES. T is used to collect edges of the final spanning tree. The set VS contains the vertex sets corresponding to the connected components of the spanning tree found so far. And ES contains, for each vertex set W in VS, an edge set $E(W)$. Initially we have $VS = \{\{v\} | v \in V\}$ and $ES = \{\{\text{all the edges incident upon } v\} | v \in V\}$. The algorithm also uses an integer parameter k, a *level function* $l: V \to \{1, 2, \ldots, k, k+1\}$, and a function

low: $V \to$ real numbers.

* Research is partially supported by NSF GJ-41538.