SIAM J. COMPUT.
Vol. 1, No. 2, June 1972

DEPTH-FIRST SEARCH AND LINEAR GRAPH ALGORITHMS*

ROBERT TARJAN†

Abstract. The value of depth-first search or "backtracking" as a technique for solving problems is illustrated by two examples. An improved version of an algorithm for finding the strongly connected components of a directed graph and an algorithm for finding the biconnected components of an undirect graph are presented. The space and time requirements of both algorithms are bounded by $k_1V + k_2E + k_3$ for some constants k_1, k_2, and k_3, where V is the number of vertices and E is the number of edges of the graph being examined.

Key words. Algorithm, backtracking, biconnectivity, connectivity, depth-first, graph, search, spanning tree, strong-connectivity.

1. Introduction.

Consider a graph G, consisting of a set of vertices $\mathscr{V}$ and a set of edges $\mathscr{E}$. The graph may either be directed (the edges are ordered pairs (v, w) of vertices; v is the *tail* and w is the *head* of the edge) or undirected (the edges are unordered pairs of vertices, also represented as (v, w)). Graphs form a suitable abstraction for problems in many areas; chemistry, electrical engineering, and sociology, for example. Thus it is important to have the most economical algorithms for answering graph-theoretical questions.

In studying graph algorithms we cannot avoid at least a few definitions. These definitions are more-or-less standard in the literature. (See Harary [3], for instance.) If $G = (\mathscr{V}, \mathscr{E})$ is a graph, a *path* $p: v \overset{*}{\Rightarrow} w$ in G is a sequence of vertices and edges leading from v to w. A path is *simple* if all its vertices are distinct. A path $p: v \overset{*}{\Rightarrow} v$ is called a *closed path*. A closed path $p: v \overset{*}{\Rightarrow} v$ is a *cycle* if all its edges are distinct and the only vertex to occur twice in p is v, which occurs exactly twice. Two cycles which are cyclic permutations of each other are considered to be the same cycle. The *undirected version* of a directed graph is the graph formed by converting each edge of the directed graph into an undirected edge and removing duplicate edges. An undirected graph is *connected* if there is a path between every pair of vertices.

A (directed rooted) *tree* T is a directed graph whose undirected version is connected, having one vertex which is the head of no edges (called the *root*), and such that all vertices except the root are the head of exactly one edge. The relation "(v, w) is an edge of T" is denoted by $v \rightarrow w$. The relation "There is a path from v to w in T" is denoted by $v \overset{*}{\rightarrow} w$. If $v \rightarrow w$, v is the *father* of w and w is a *son* of v. If $v \overset{*}{\rightarrow} w$, v is an *ancestor* of w and w is a *descendant* of v. Every vertex is an ancestor and a descendant of itself. If v is a vertex in a tree T, T_v is the subtree of T having as vertices all the descendants of v in T. If G is a directed graph, a tree T is a *spanning tree* of G if T is a subgraph of G and T contains all the vertices of G.

If R and S are binary relations, R^* is the transitive closure of R, R^{-1} is the inverse of R, and

$$RS = \{(u, w) \mid \exists v((u, v) \in R \;\&\; (v, w) \in S)\}.$$

* Received by the editors August 30, 1971, and in revised form March 9, 1972.

† Department of Computer Science, Cornell University, Ithaca, New York 14850. This research was supported by the Hertz Foundation and the National Science Foundation under Grant GJ-992.

If $f, f_1, \cdots, f_n$ are functions of $x_1, \cdots, x_n$, we say f is $O(f_1, \cdots, f_n)$ if

$$|f(x_1, \cdots, x_n)| \leqq k_0 + k_1|f_1(x_1, \cdots, x_n)| + \cdots + k_n|f_n(x_1, \cdots, x_n)|$$

for all x_i and some constants $k_0, \cdots, k_n$. We shall assume a random-access computer model.

2. Depth-first search. Backtracking, or depth-first search, is a technique which has been widely used for finding solutions to problems in combinatorial theory and artificial intelligence [2], [11] but whose properties have not been widely analyzed. Suppose G is a graph which we wish to explore. Initially all the vertices of G are unexplored. We start from some vertex of G and choose an edge to follow. Traversing the edge leads to a new vertex. We continue in this way; at each step we select an unexplored edge leading from a vertex already reached and we traverse this edge. The edge leads to some vertex, either new or already reached. Whenever we run out of edges leading from old vertices, we choose some unreached vertex, if any exists, and begin a new exploration from this point. Eventually we will traverse all the edges of G, each exactly once. Such a process is called a *search* of G.

There are many ways of searching a graph, depending upon the way in which edges to search are selected. Consider the following choice rule: when selecting an edge to traverse, always choose an edge emanating from the vertex most recently reached which still has unexplored edges. A search which uses this rule is called a *depth-first search*. The set of old vertices with possibly unexplored edges may be stored on a stack. Thus a depth-first search is very easy to program either iteratively or recursively, provided we have a suitable computer representation of a graph.

DEFINITION 1. Let $G = (\mathscr{V}, \mathscr{E})$ be a graph. For each vertex $v \in \mathscr{V}$ we may construct a list containing all vertices w such that $(v, w) \in \mathscr{E}$. Such a list is called an *adjacency list* for vertex v. A set of such lists, one for each vertex in G, is called an *adjacency structure* for G.

If the graph G is undirected, each edge (v, w) is represented twice in an adjacency structure; once for v and once for w. If G is directed, each edge (v, w) is represented once: vertex w appears in the adjacency list of vertex v. A single graph may have many adjacency structures; in fact, each ordering of the edges around the vertices of G gives a unique adjacency structure, and each adjacency structure corresponds to a unique ordering of the edges at each vertex. Using an adjacency structure for a graph, we can perform depth-first searches in a very efficient manner, as we shall see.

Suppose G is a connected undirected graph. A search of G imposes a direction on each edge of G given by the direction in which the edge is traversed when the search is performed. Thus G is converted into a directed graph G'. The set of edges which lead to a new vertex when traversed during the search defines a spanning tree of G'. In general, the arcs of G' which are not part of the spanning tree interconnect the paths in the tree. However, if the search is depth-first, each edge (v, w) not in the spanning tree connects vertex v with one of its ancestors w.

DEFINITION 2. Let P be a directed graph, consisting of two disjoint sets of edges, denoted by $v \rightarrow w$ and $v \dashrightarrow w$ respectively. Suppose P satisfies the following properties:

(i) The subgraph T containing the edges $v \rightarrow w$ is a spanning tree of P.