BIT 28 (1988), 353–356

SCIENTIFIC NOTES

AN OPTIMAL TIME ALGORITHM FOR FINDING A MAXIMUM WEIGHT INDEPENDENT SET IN A TREE

G. H. CHEN [1], M. T. KUO [1] and J. P. SHEU [2]

[1]) *Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, Republic of China*

[2]) *Department of Electrical Engineering, National Central University, Chung-Li, Taiwan, Republic of China*

Abstract.

The maximum weight independent set problem for a general graph is NP-hard. But for some special classes of graphs, polynomial time algorithms do exist for solving it. Based on the divide-and-conquer strategy, Pawagi has presented an $O(|V|\log|V|)$ time algorithm for solving this problem on a tree. In this paper, we propose an $O(|V|)$ time algorithm to improve Pawagi's result. The proposed algorithm is based on the dynamic programming strategy and is time optimal within a constant factor.

CR categories: G.2.2, F.2.2.

Keywords and Phrases: maximum weight independent set, dynamic programming.

1. Introduction.

Let $G = (V, E)$ be an undirected finite graph where V denotes the set of vertices and E denotes the set of edges. If G is connected and acyclic, then it is called a tree. A subset I of V is called an independent set of G if no two vertices of I are adjacent in G. Assume that a positive weight $w(i)$ is associated with each vertex i. We define the weight $w(I)$ of an independent set I to be the sum of the weights of all the vertices in I. That is, $w(I) = \sum_{i \in I} w(i)$. Further, an independent set is called a maximum weight independent set if it has maximum weight. Finding a maximum weight independent set in a general graph is NP-hard [1]. But for some special classes of graphs [1], [2], [3] this problem is likely to be in P. Based on the divide-and-conquer strategy, Pawagi [3] has presented an $O(|V|\log|V|)$ time algorithm to find a maximum weight independent set in a tree, where $|V|$ denotes the cardinality of V. In this paper, we shall improve Pawagi's result. Our proposed algorithm is based on the dynamic programming strategy and needs only $O(|V|)$ time.

Received November 1987. Revised February 1988.

2. An $O(|V|)$ time algorithm.

Let G be a tree and let T_i denote the subtree of G which is rooted at the vertex i. We define $M(i)$ and $M'(i)$ for each vertex i as follows.

$$M(i) = \max\{w(I)|I \text{ is an independent set in } T_i \text{ and } i \in I\}$$

$$M'(i) = \max\{w(I)|I \text{ is an independent set in } T_i \text{ and } i \notin I\}.$$

$M(i)$ and $M'(i)$ are the maximum weights of the independent sets in T_i that contain i, and do not contain i, respectively. These two values can be computed recursively by the following two equations:

$$M(i) = w(i) + \sum_j M'(j) \quad \text{and} \quad M'(i) = \sum_j \max\{M(j), M'(j)\},$$

where j is a child of i. Note that $M(i) = w(i)$ and $M'(i) = 0$ for each leaf vertex i. In Figure 1, we give an example of a tree and the values of $w(i)$, $M(i)$, and $M'(i)$ for each vertex i are listed in Table 1.

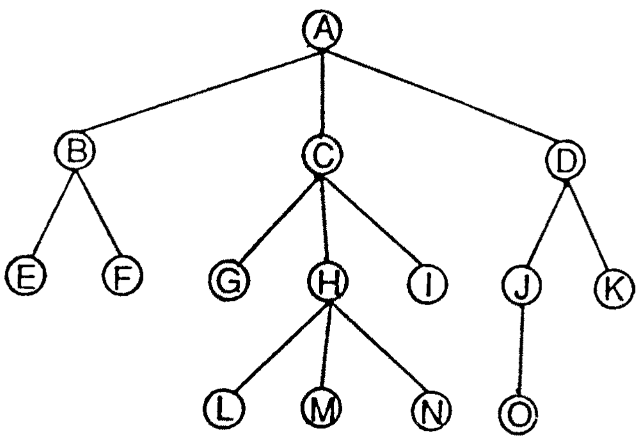

Fig. 1. An example of a tree.

After computing $M(i)$ and $M'(i)$ for each vertex i, we can find a maximum weight independent set I by examining G, starting from the root and going down to the leaves. The root is included in I if $M(\text{root}) > M'(\text{root})$ and a nonroot vertex j is included in I if its parent is not in I and $M(j) > M'(j)$. For example, the maximum weight independent set for the example of Figure 1 is {A, E, F, G, L, M, N, I, J, K}.

The following is an informal description of our algorithm.

Table 1. *Data for the example of Figure* 1.

i	$w(i)$	$M(i)$	$M'(i)$
A	6	53	50
B	4	4	11
C	8	23	20
D	8	10	16
E	5	5	0
F	6	6	0
G	2	2	0
H	8	8	15
I	3	3	0
J	9	9	2
K	7	7	0
L	5	5	0
M	4	4	0
N	6	6	0
O	2	2	0

Algorithm. Find a maximum weight independent set in a tree.

phase 1. Compute $M(i)$ and $M'(i)$ for each vertex i, starting from the leaves and going up to the root.

step 1. For each leaf vertex j,
$M(j) \leftarrow w(j)$ and $M'(j) \leftarrow 0$.

step 2. For each nonleaf vertex i,
$M(i) \leftarrow w(i) + \sum_j M'(j)$ and
$M'(i) \leftarrow \sum_j \max\{M(j), M'(j)\}$, where j is a child of i.

phase 2. Find a maximum weight independent set by examining the tree, starting from the root and going down to leaves.

step 1. $I \leftarrow \emptyset$.

step 2. If $M(\text{root}) > M'(\text{root})$, $I \leftarrow I + \{\text{root}\}$.

step 3. For each nonroot vertex j,
$I \leftarrow I + \{j\}$, if parent of j is not included in I and $M(j) > M'(j)$.

In *phase* 1, both $M(i)$ and $M'(i)$ are computed for each vertex i. Also, each vertex is examined exactly once in *phase* 2. Thus, the time complexity of the proposed algorithm is $O(|V|)$.

3. Concluding remarks.

In this paper we have proposed an $O(|V|)$ time algorithm for solving the maximum weight independent set problem on a tree. Since there are at most $|V|-1$ vertices in the independent set, this problem has a lower bound of

$\Omega(|V|)$ time. Therefore, our proposed algorithm is time optimal within a constant factor.

REFERENCES

1. M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, CA., 1979.
2. A. Frank, *Some polynomial algorithms for certain graphs and hyper-graphs*, in Proc. of the fifth British Combin. Conference, 1975, 211–226.
3. S. Pawagi, *Maximum weight independent set in trees*, BIT 27, 2 (1987), 170–180.