JOURNAL OF ALGORITHMS 5, 48–59 (1984)

Scheduling Precedence Graphs of Bounded Height

DANNY DOLEV

IBM Research Laboratory, San Jose, California 95193

AND

MANFRED K. WARMUTH

Computer Science Department, University of California, Santa Cruz, California 95064

Received May 2, 1982; revised August 16, 1982

The existence of a schedule for a partially ordered set of unit length tasks on m identical processors is known to be NP-complete (J. D. Ullman, NP-complete scheduling problems, *J. Comput. System Sci.* **10** (1975), 384–393). The problem remains NP-complete even if we restrict the precedence graph to be of height bounded by a constant. (J. K. Lenkstra and A. H. G. Rinnooy Kan, Complexity of scheduling under precedence constraints, *Operations Res.* **26** (1978), 22–35; D. Dolev and M. K. Warmuth, "Scheduling Flat Graphs," IBM Research Report RJ 3398, 1982). In these NP-completeness proofs the upper bound on the number of available processors varies with the problem instance. We present a polynomial algorithm for the case where the upper bound on the number of available processors and the height of the precedence graph are both constants.

1. INTRODUCTION

The goal of deterministic scheduling is to obtain efficient algorithms under the assumption that all the information about the tasks to be scheduled is known in advance. One of the fundamental problems in deterministic scheduling is to schedule a collection of partially ordered, unit-execution-time (UET) tasks on a number of identical processors. As in [6] and [4], we allow the number of available processors to vary with time.

A UET scheduling problem is given by a precedence graph and a profile. The *precedence graph* is a directed acyclic graph (dag) that specifies the precedence constraints between the UET tasks. A *profile* is a sequence of

natural numbers specifying how many processors are available in the first, second, ... time slot. A profile is *straight* if there is the same number of processors available in all time slots. Given a dag and a profile, we are interested in *schedules* for the dag that *fit* the profile, meaning we never schedule more tasks than there are processors available. The *breadth* of a profile is the upper bound on the number of processors that are available at any time slot of the profile. For example, the profile of Table 1 is of breadth three.

Various aspects of scheduling theory have been studied in recent years [1, 7], and many scheduling problems are known to be NP-complete. In all known NP-completeness results of UET scheduling, the profile has *unbounded breadth*, meaning the breadth of the profile varies with different problem instances. A profile is called *straight* if it has the same number of processors available in every time slot. If the number of processors is allowed to vary with time, we call the profile *variable*.

The first NP-completeness result about UET scheduling was published by Ullman [9]. He showed that to decide if there exists a schedule for a profile of a fixed length is NP-complete, if the precedence graph is arbitrary, and the profile is of unbounded breadth and straight. Lenkstra and Rinnooy Kan [8] proved that the above decision problem remains NP-complete even if the height of the precedence graph is one, and the profile contains three time units with an arbitrary number of available processors in each. Even if we restrict the number of time units with an arbitrary number of processors to one, the existence of schedule is NP-complete [3].

We prove that if the breadth of the profile is bounded by a constant, say m, and the height of the graph is bounded by a constant h, then there exists an $O(n^{h(m-1)+1})$ algorithm to find an optimal schedule. The number of available processors in the profiles we consider varies with time. This is important because the simple approach of making the profile straight by adding a "padding" component to the graph does not work. In general, such a component would not be of bounded height.

We first find a property that some optimal schedule has, and show that only polynomially-many schedules have that property. Since our algorithm uses dynamic programming to scan all possible schedules having that property, the algorithm actually finds an optimal schedule. The algorithm uses recursion on the height of the precedence graph, and therefore provides a polynomial algorithm only for graphs of bounded height. But for specific classes of graphs, one can use recursion on the structure of the graph, and similarly obtain polynomial algorithms. Warmuth [10] uses such a technique, combined with the Elite Theorem [3], to obtain polynomial algorithm for scheduling Series-Parallel graph with bounded decomposition.

The main question which remains open is that of scheduling UET on a constant number of processors subjected to precedence graph of arbitrary height. Clearly, it is enough to solve the problem for straight profiles. The case where the breadth of the profile is two has a polynomial solution [2, 5].

Note that if the breadth of the profile is arbitrary the problem is NP-complete even for precedence graphs (partial order) of a special form [6, 10]. On the other hand, polynomial algorithms have been published [8, 10, 3, 4] for scheduling the same classes of precedence graphs on profiles of fixed breadth.

2. Definitions and Notations

A *precedence graph* is a directed acyclic graph (*dag*) $G = (V, E)$, where V is the set of n *vertices* (or *tasks*) and E the set of *edges* of G. A precedence graph G specifies the precedence constraints between the vertices (tasks) of G. We assume that if a task x has to be executed before a task y, then there exists a (directed) path from x to y in G.

If there exists a path from x to y, then x is a *predecessor* of y, and y is a *successor* of x. In the case where the longest path from a vertex x to a vertex y is the edge (x, y), x is an *immediate predecessor* of y and y is an *immediate successor* of x. A vertex with no predecessors is *initial.*

By $h(G)$ we mean the *height* of G, which is the length of the longest path in G. For a vertex $x \in G$ (i.e., $x \in V$) we denote by $h(x)$ the length of the longest path that starts at x. A vertex with no successors has zero height. Vertices with identical height are said to be at the same *level.*

The graph $G' = (V', E')$ is a subgraph of G, if

(i) $V' \subseteq V$;

(ii) for all x and y in V', x is a predecessor of y in G' iff x is a predecessor of y in G.

We partition the time scale into time slots of length one. The time interval $[i - 1, i)$ for $i \geqslant 1$ is the ith time slot. A *profile* is a sequence of positive integers specifying the number of identical processors that are available in each time slot. We shall interpret profile $M = (m_1, \ldots, m_d)$, where d is its length, to mean that for each slot i in $[0, d)$ there are m_i processors available.

A *schedule* for S for a precedence graph G is a sequence of sets $(S)_1, \ldots, (S)_k$ such that:

(i) the sets $(S)_i$, for $1 \leqslant i \leqslant k$, partition the vertices of G;

(ii) if $x \in (S)_i$ and $y \in (S)_j$, for $1 \leqslant i \leqslant j \leqslant k$, then there is no directed path from y to x.

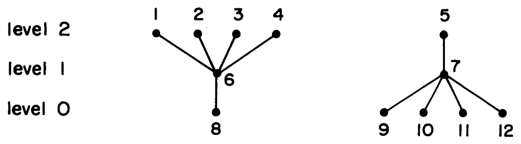

FIGURE 1

The *length* of a schedule is the index of the last nonempty set in the sequence. A minimum length schedule is called *optimal*. The schedule S *fits* the profile M if the length of S is not greater than the length of the profile and the cardinality of $(S)_i$ is not greater than m_i. The set of tasks $(S)_i$ gets executed in the ith time slot, that is $|(S)_i|$ of the m_i processors of slot i get assigned a task of $(S)_i$ during the time interval $[i-1, i)$. Note that all the tasks have unit length which corresponds to the length of a time slot.

For example, assume we have a set of twelve tasks subjected to the precedence graph G presented in Fig. 1. We name the tasks by integer numbers, and assume that the graph is directed downwards. We look for a schedule for G that fits the profile $M = (2,3,3,1,3,2)$. The following sequence S is a valid schedule:

$$\{1,2\};\{3,4,5\};\{6,7\};\{8\};\{9,10,11\};\{12\}.$$

The sequence S and the profile M can be represented as in Table 1. The length of the schedule S is six, but there exists a schedule S' of length 5 for G and M, as we see in Table 2. The schedule S is clearly not optimal whereas S' is optimal.

Let G be a dag and M be a profile. The following notations are used throughout the rest of the paper.

The schedule $FST(k, S)$ consists of the first k slots of S, that is, $FST(k, S)\colon = (S)_1, (S)_2, \dots, (S)_k$. The subschedule of S starting at the kth slot is defined by: $REM(k, S)\colon = (S)_k, \dots, (S)_{l(s)}$. We denote by $FST(k, S, G)$ and $REM(k, S, G)$ the subdags of G induced by the vertices in the schedules $FST(k, S)$ and $REM(k, S)$, respectively.

Let $(S)_z$ be the first slot of S containing a vertex of height zero. We define $(S)_z$ to be the *zero-slot* and z the *zero-index* of S. The zero-slot $(S)_z$ is

TABLE 1

Slot	1	2	3	4	5	6
P_1	1	3	6	8	9	12
P_2	2	4	7		10	
P_3		5			11	
m_i	2	3	3	1	3	2

TABLE 2

Slot	1	2	3	4	5	6
P_1	1	2	4	6	8	
P_2	5	3	9		11	
P_3		7	10		12	
m_i	2	3	3	1	3	2

partitioned into two sets, the *high-set* and the *zero-set*. The high-set (respectively, zero-set) of S consists of all vertices of $(S)_z$ of positive (respectively zero) height. An illustration of the above definitions is given in Fig. 2.

DEFINITION. Let z be the zero-index of the schedule S. Then S is *zero-adjusted* if the following three conditions hold:

Z1: If there exists an idle period in the zero-slot of S, then the zero-slot contains all initial vertices of $REM(z, S, G)$.

Z2: Optimal schedules for $FST(z-1, S, G)$ fitting M have length $z-1$.

Z3: The high-set of S is the set of all initial vertices of $REM(z, S, G)$ having positive height.

3. Existence of Zero-Adjusted Optimal Schedules

Our algorithm constructs an optimal zero-adjusted schedule. We show that such a schedule always exists.

THEOREM 1. *There exists an optimal, zero-adjusted schedule.*

Proof. Let G be a dag and $M = (m_1, \ldots, m_d)$ be a profile. Denote by l the length of an optimal schedule for G that fits M. For each optimal schedule S that fits M, let $a_S = (i_1, \ldots, i_l)$ be the vector such that i_j is the number of idle periods in the jth time slot of the schedule S.

We define the following order between two vectors $a_S = (i_1, \ldots, i_l)$ and $a_{S'} = (i'_1, \ldots, i'_l)$: $a_S < a_{S'}$, if there exists r such that $i_r < i'_r$ and $i_k = i'_k$, for all k, where $1 \leqslant r < k \leqslant l$. That is, the vectors are ordered lexicographically according to their indices from right to left.

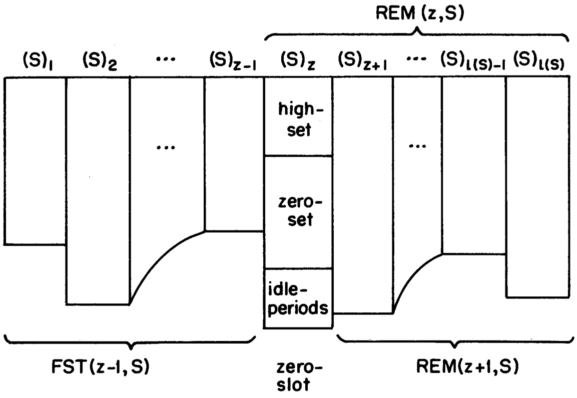

FIGURE 2

Choose A to be the set of all optimal schedules for G that fit M and have the maximum vector a. Choose B to be the subset of A containing all schedules with maximum zero-index. Finally, choose C to be the subset of B which consists of all schedules having a maximum cardinality zero-set. The set C is clearly nonempty and, the following four claims show that all schedules of C are zero-adjusted.

Claim 1. *Every schedule in A fulfills condition Z1.*

Z1: If there exists an idle period in the zero-slot of S, then the zero-slot contains all initial vertices of $REM(z, S, G)$.

Proof. Assume to the contrary that *Z1* does not hold for a schedule S in A. Thus S contains an idle period in its zero-slot $(S)_z$, but it does not contain all initial vertices of $REM(z, S, G)$. Let u be an initial vertex of $REM(z, S, G)$ that is not scheduled in the zero-slot of S. Assume u is scheduled in slot t of S, for some t, $z < t \leqslant l(S)$. By moving the vertex u from slot t to the zero-slot of S we construct a new schedule S' for G fitting M. Moving u to the zero-slot does not violate any precedence constraints specified by G, since u is initial in $REM(z, S, G)$. Therefore, S' is a valid schedule for G fitting M. Moving u to an earlier slot clearly did not increase the length of the schedule; thus $l(S') \leqslant l(S)$, and the optimality of S implies the optimality of S'.

Let $a_S = (i_1, \ldots, i_l)$ and $a_{S'} = (i'_1, \ldots, i'_l)$ it is easy to see that $i'_j = i_j$, for $j \neq z$, $j \neq t$, $1 \leqslant j \leqslant l$, and $i'_z = i_z - 1$ and $i'_t = i_t + 1$. This implies that $a_S < a_{S'}$, which contradicts the fact that $S \in A$, since a_S is not maximum.

Claim 2. *Every schedule in A fulfills condition Z2.*

Z2: Optimal schedules for $FST(z - 1, S, G)$ fitting M have length $z - 1$.

Proof. Let S be a schedule of A with zero-index z and assume to the contrary that *Z2* does not hold.

Case $(S)_{z-1} = \phi$. In this case we move an initial vertex u of $REM(z, S, G)$ to slot $z - 1$. Note that the number of processors available at any slot is greater than zero. As in the previous claim, by moving u from a later slot to slot $z - 1$ we can construct a schedule S' which is optimal and has a bigger vector than S. This contradicts the fact that S is in A, since it does not have a maximum vector.

Case $(S)_{z-1} \neq \phi$. In this case the length of $FST(z - 1, S)$ is $z - 1$. Since *Z2* does not hold for S, there exists a schedule $\overline{S}$ for $FST(z - 1, S, G)$ fitting M of length less than $z - 1$. We replace the subschedule $FST(z -$

1, S) of S by the schedule $\bar{S}$ to construct a new schedule S^* for G and M as follows:

$$(S^*)_i := (S')_i, \text{ for } 1 \leqslant i \leqslant l(\bar{S})$$
$$(S^*)_i := \phi, \text{ for } l(\bar{S}) < i < z$$
$$(S^*)_i := (S)_i, \text{ for } z \leqslant i < l(S).$$

Let $a_S = (i_1, \ldots, i_l)$ and $a_{S^*} = (i_1^*, \ldots, i_l^*)$ be the corresponding vectors of the schedules S and S^*, respectively. By the definition of S and S^* we know that

$$i_j = i_j^*, \text{ for } z \leqslant j \leqslant l(S)$$
$$i_{z-1} < m_{z-1}$$
$$i_{z-1}^* = m_{z-1}.$$

Thus $i_{z-1} < i_{z-1}^*$ and $a_S < a_{S^*}$. This again contradicts the fact that S is in A, and the proof of Claim 2 is completed.

CLAIM 3. *Every schedule in B, whose zero-set consists of one vertex, fulfills Z3.*

Z3: The high-set of S is the set of all initial vertices of $REM(z, S, G)$ having positive height.

Proof. Let S be a schedule of B, whose zero-set consists of one vertex u, and for which *Z3* does not hold. This implies that $REM(z, S, G)$ has an initial vertex w of positive height, which is not scheduled in the zero-slot z of S. Since both u and w are initial in $REM(z, S, G)$ and u does not have any successors, we can exchange u with w. Call the new schedule S'. Clearly, exchanging u with w did not affect the idle periods. Therefore, since S is in A, S' is also.

Note that $(S')_z$ does not have any more vertices of height zero. Thus the zero-index of S' is higher than the zero-index of S. This contradicts the assumption that S is in B.

CLAIM 4. *Every schedule of C, whose zero-set consists of more than one vertex, fulfills Z3.*

Proof. Let S be a schedule of C, whose zero-set consists of more than one vertex, and for which *Z3* does not hold. Let u be a vertex of height zero (zero-vertex) of the zero-slot (S_z) of S. Since *Z3* does not hold for S, $REM(z, S, G)$ contains an initial vertex w of positive height that is not scheduled in the zero-slot of S.

As in the previous claim, we can exchange u with w and the resulting schedule S' is still in A. Note that $(S)_z$ has one zero-vertex less than $(S')_z$.

Since $(S)_z$ contains more than one zero vertex, we conclude that $(S')_z$ contains at least one zero vertex. Thus S and S' have the same zero-index and $S \in B$ implies $S' \in B$. This contradicts the assumption that $S \in C$, since $S' \in B$ and the zero-set of S' is one smaller than the zero-set of S.

Applying the above four claims we conclude that every schedule of C is zero adjusted, which completes the proof of the theorem. □

In the next section we present a polynomial algorithm for finding an optimal, zero-adjusted schedule for the case where the height of the dag and the breadth of the profile are bounded by constants. The following theorem is needed to prove the correctness of this algorithm.

Let T be a subset of the vertices of G, then CLOSE(T) is the subgraph of G containing all the successors of vertices from T.

THEOREM 2. *Let S be an optimal, zero-adjusted schedule for a precedence graph G fitting a profile M of breadth m, and let z be the zero-index of S.*

(*i*) *The high-set of S determines $FST(z-1, S, G)$, the zero-index of S, the cardinality of zero-set of S, the length of S, and up to isomorphism it determines $REM(z+1, S, G)$.*

(*ii*) *The high-set of S has at most $m-1$ vertices and there are $O(n^{m-1})$ possible choices for the high-set.*

(*iii*) *If $h(G) > 0$, then $h(FST(z-1, S, G)) \leqslant h(G) - 1$ and $h(REM(z+1, S, G)) \leqslant h(G) - 1$.*

Proof of (i). Let H be the high-set, z be the zero-index and Z be the zero-set of S. The condition *Z3* of the definition of zero-adjusted schedule implies that $FST(z-1, S, G)$ in the subdag obtained from G after removing the vertices of CLOSE(H) and all vertices of height zero. Thus $FST(z-1, S, G)$ is determined by H.

Since *Z2* implies that $z-1$ is the length of an optimal schedule for $FST(z-1, S, G)$ and M, the zero index z is also determined by H.

The dag $REM(z, S, G)$ is the subdag of G that remains after removing the vertices of $FST(z-1, S, G)$. Since $FST(z-1, S, G)$ is determined by H, so is the dag $REM(z, S, G)$.

Assume there are m_z processors available in slot z of M. If $REM(z, S, G)$ contains at most $m_z - |H|$ initial vertices of height zero, then Z consists of all initial vertices of $REM(z, S, G)$ having height zero (see Condition *Z1*). Otherwise, Z is a set of some $m_z - |H|$ initial vertices of $REM(z, S, G)$ having height zero. Since $REM(z, S, G)$ is determined by H, the cardinality of Z is also.

We want to show that $REM(z+1, S, G)$ is determined by H up to isomorphism. Let n_z be the number of initial vertices of $REM(z, S, G)$ having height zero. Then $REM(z+1, S, G)$ is the subdag of G: containing all the successors of H (not including H), plus $n_z - |Z|$ initial vertices of height zero. All initial vertices of $REM(z, S, G)$ of height zero do not have any predecessors nor any successor; therefore, they are isomorphic to each other. Since $REM(z, S, G)$ and $|Z|$ are determined by H, $REM(z + 1, S, G)$ is determined by H up to isomorphism.

Finally, we show that $l(S)$ is determined by H. Note that $l(S)$ equals z plus the length of $REM(z + 1, S)$. Using the fact that the zero-index z is determined by H we only have to show that the length of $REM(z + 1, S)$ is also. To do this recall that $REM(z + 1, S, G)$ is determined by H up to isomorphism, and observe that the optimality of S for G and M implies the optimality for $REM(z + 1, S, G)$ and $(m_{z+1}, \ldots, m_d)$.

This completes the proof that the decomposition of S as shown in Fig. 2 is determined by its high-set.

Proof of (ii). Since the profile M has breadth m we have $m_z \leqslant m$. Together with the fact that $|H| + |Z| \leqslant m_z$ and $|Z| \geqslant 1$, we conclude that $|H| \leqslant m - 1$. Note that the high set H of S might be empty.

There are $O(n^k)$ subsets of exactly k vertices of G. Using the fact that $|H| \leqslant m - 1$, it follows that there is $O(\sum_{k=0}^{m-1} n^k)$ choices for the high-set of a zero-adjusted schedule. Therefore we conclude that there are $O(n^{m-1})$ choices for the high-set of S.

Proof of (iii). $h(FST(z - 1, S, G)) \leqslant h(G) - 1$ follows from the fact that $FST(z - 1, S, G)$ does not contain any vertices of G having height zero. Since $h(G) > 0$, the longest path in $FST(z - 1, S, G)$ is at least one shorter than the longest path in G.

To show the second inequality, observe that $h(REM(z, S, G)) \leqslant h(G)$. The case $h(REM(z, S, G)) = 0$ is trivial. Otherwise, $h(REM(z, S, G)) > 0$, and by condition *Z3* we know that the "top layer" of the subdag $REM(z, S, G)$ has been removed and scheduled in $(S)_z$. We conclude that $h(REM(z + 1, S, G)) = h(REM(z, S, G)) - 1$, which implies that $h(REM(z + 1, S, G)) \leqslant h(G) - 1$. This completes the proof of Theorem 2. □

4. The Algorithm

We are ready now to present the algorithm for finding optimal zero-adjusted schedule for a given precedence graph and profile.

Theorem 3. *Let G be an arbitrary dag and M be a profile of breadth m. Then an optimal, zero-adjusted schedule for G that fits M can be found in time $O(n^{h(G)(m-1)+1})$.*

Proof. By Theorem 1, an optimal zero-adjusted schedule always exists. The following algorithm finds such a schedule S for G that fits M. The algorithm recurses on the height of the dag. Let $M = (m_1,\ldots,m_d)$. We choose $d = n$ to assure feasibility.

Algorithm

```
Procedure $R(G, M, S)$:
  1. *if* $h(G) = 0$ *then*
     1.1 $k: = 0$
     1.2. *while* $G$ in nonempty *do*
          1.2.1. $k: = k + 1$
          1.2.2. Put vertices from $G$ into $S$ until $(S)_k = m_k$ or
                 $G$ is empty
     1.3. *return*
  2. $l: = n$
  3. *for* all possible choices of the high-set $H$ *do*
     3.1. Find the subdag $\bar{G}$ of $G$ obtained by removing all vertices of
CLOSE($H$)
          and all vertices of height zero from $G$.
     3.2. $R(\bar{G}, M, \bar{S})$
     3.3. *if* $|H| \leqslant m_{l(\bar{S})+1} - 1$, *then*
          3.3.1. Find the subdag $G^*$ of $G$ obtained by removing the
                 vertices of $\bar{G}$ from $G$.
          3.3.2. Determine $n_z$, the number of initial vertices of $G^*$ of
                 height zero.
          3.3.3. Choose $\min(n_z, m_{l(\bar{S})+1} - |H|)$ initial vertices of $G^*$ hav-
                 ing height zero to be the zero-set $Z$.
          3.3.4. Find the subdag $G'$ of $G^*$ obtained by removing the
                 vertices of $H$ and $Z$ from $G^*$.
          3.3.5. $R(G',(m_{l(\bar{S})+2},\ldots,m_d), S')$
          3.3.6. *if* $l(S) + 1 + l(S') < l$, *then* $l = l(S) + 1 + l(S')$ and
                 $S: = \bar{S}||\{H \cup Z\}||S'$,
                  where || denotes the concatenation of schedules.
```

Proof of correctness. If $h(G) = 0$, then in Step 1 the algorithm clearly finds an optimal schedule for G fitting every profile M. Assume that the algorithm returns an optimal schedule for any dag I such that $h(I) < h$, for some $h > 0$, fitting any profile M. Using this inductive hypothesis we want to show that the algorithm returns an optimal schedule for G fitting M, if $h(G) = h$.

Theorem 1 assures the existence of an optimal, zero-adjusted schedule for G fitting M. Theorem 2 says that a zero-adjusted schedule is determined by its high-set (details are given in the proof of Statement (i) of Theorem 2). In Step 3 we recursively find for every high-set H a minimum length, zero-adjusted schedule whose high-set is H. In the recursive calls at Steps 3.2 and 3.3.5 we make use of the inductive hypothesis. Note that by Theorem 2, Statement (iii), $h(\overline{G})$, and $h(G')$ have height at most $h(G) - 1 = h - 1$. In Step 3.3.6 the algorithm finds a minimum length, zero-adjusted schedule for G fitting M, which is optimal by Theorem 1. This completes the proof of correctness.

Proof of the time bound. The time for the algorithm $R(G, M, S)$ is denoted as $T(n, m, h(G))$. It satisfies the following recursion.

$$T(n, m, 0) = c_1 \cdot n$$

$$T(n, m, h(G)) < n^{(m-1)} \cdot \left(2 \cdot T(n, m, h(G) - 1) + c_2 \cdot n^2\right) \text{ if } h(G) > 0.$$

The fact that $T(n, m, 0) = c_1 \cdot n$ is trivial (see Step 1). To show the remaining we observe that there are $O(n^{m-1})$ choices for the high-set (see Theorem 2, Statement (ii)). Once the high-set is chosen we call R twice recursively in Steps 3.2 and 3.3.5. Each of these calls costs less than $T(n, m, h(G) - 1)$ since $|\overline{G}| < n$, $|G'| < n$ and by Theorem 2, Statement (iii), $h(\overline{G}) \leqslant h(G) - 1$ and $h(G') \leqslant h(G) - 1$. All the remaining steps in Loop 3 cost $c_2 \cdot n^2$, if we assume reasonable data structures for G and S.

Since we can assume $m \geqslant 2$, we have $T(n, m, h(G)) \geqslant n^2$ if $h(G) > 0$. Using this inequality we can simplify our formula for $T(n, m, h(G))$:

$$T(n, m, 0) = c_1 \cdot n$$

$$T(n, m, h(G)) < c_3 \cdot n^{m-1} \cdot T(n, m, h(G) - 1), \text{ if } h(G) > 0,$$

where c_3 is a new constant. Evaluating the above formulas we get the time bound of $O(n^{h(G)(m-1)+1})$. □

ACKNOWLEDGMENT

We would like to thank Barbara Simons for many useful suggestions and thorough proofreading of this paper.

REFERENCES

1. E. G. COFFMAN (Ed.), "Computer and Job Shop Scheduling Theory," pp. 54–59, Wiley, New York, 1976.
2. E. G. COFFMAN, JR., AND R. L. GRAHAM, Optical scheduling for two-processors systems, *Acta Informatica* **1** (1972), 200–213.

3. D. Dolev and M. K. Warmuth, "Scheduling Flat Graphs," IBM Research Report RJ3398, 1982.
4. D. Dolev and M. K. Warmuth, "Profile Scheduling of Opposing Forests and Level Orders," IBM Research Report RJ3553, 1982.
5. H. N. Gabow, An almost linear algorithm for two processor scheduling, *Assoc. Comput. Mach.*, in press. See also Technical Report CU-169-80, Department of Computer Science, University of Colorado, Boulder, January 1980.
6. M. R. Garey, D. S. Johnson, R. E. Tarjan, and M. Yannakakis, Scheduling opposing forests, *SIAM J. Algebraic and Discrete Methods*, in press.
7. R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: A survey, *Ann. Discrete Math.* **5** (1979), 287–326. See also Bell Laboratories, TM-79-1216-12, 1979.
8. J. K. Lenkstra and A. H. G. Rinnooy Kan, Complexity of scheduling under precedence constraints, *Operations Res.* **26** (1978) 22–35.
9. J. D. Ullman, NP-Complete scheduling problems, *J. Comput. System Sci.* **10** (1975), 384–393.
10. M. K. Warmuth, "Scheduling on Profiles of Constant Breadth," Ph.D. Thesis, Department of Computer Science, University of Colorado, Boulder, Colorado, August 1981.