An Optimal Algorithm for Assigning Cryptographic Keys to Control Access in a Hierarchy

STEPHEN J. MacKINNON, PETER D. TAYLOR, HENK MEIJER, AND SELIM G. AKL, SENIOR MEMBER, IEEE

Abstract—**A cryptographic scheme for controlling access to information within a group of users organized in a hierarchy was proposed in [1]. The scheme enables a user at some level to compute from his own cryptographic key the keys of the users below him in the organization.**

In such a system there exists the possibility of two users collaborating to compute a key to which they are not entitled. This paper formulates a condition which prevents such cooperative attacks and characterizes all key assignments which satisfy the condition.

The key generation algorithm of [1] is infeasible when there is a large number of users. This paper discusses other algorithms and their feasibility.

Index Terms—**Access control, canonical assignment, cooperative attack, cryptographic key, hierarchy, key generation algorithm, partially ordered set.**

I. INTRODUCTION

A scheme based on cryptography was proposed in [1] for controlling access to information in an organization where hierarchy is represented by a poset. An algorithm was given which enables a member of the organization at some level of the hierarchy to derive from his own cryptographic key the keys of members below him in the hierarchy, and consequently to have access to information enciphered under those keys. Another important property of the algorithm is that it provides security against two or more users of the system collaborating to compute a key to which they are not entitled.

The purpose of this paper is first to show that the key generation algorithm of [1] becomes inefficient when the number of users is large, and then to describe an improved algorithm and discuss its optimality.

II. CRYPTOGRAPHY AND HIERARCHY ACCESS

Assume a communication system where every user belongs to one of a number of disjoint security classes U_i, $i \in S$, and periodically receives data from an authority U_0. The set of classes is partially ordered by the relation $\leq$ where $U_i \leq U_j$ for i, j in S means that users in U_j can have access to information destined to users in U_i. By definition, every class U_i in the set is such that $U_i \leq U_0$. The problem is to design a scheme such that an object x broadcast by U_0 and addressed to users in U_m is accessible to users in U_i if and only if $U_m \leq U_i$.

The cryptographic solution to this problem presented in [1] goes as follows. The authority U_0 generates a set of keys $\{K_i\colon i \in S\}$ and distributes K_i (secretly) to all users in any U_j for which $U_i \leq U_j$. When U_0 desires to broadcast a message x for U_m, it first enciphers it under K_m to obtain

$$x' = E(K_m, x)$$

and then broadcasts $[x', m]$. Only users in possession of K_m will be able to retrieve x from

$$x = D(K_m, x').$$

An important advantage of this solution is that it requires only one copy of the data object x to be stored or broadcast (in enciphered form). As pointed out in [1], however, its disadvantage is the large number of keys held by each user. The worst case occurs when some U_j is a maximum element and users in U_j have to store the keys of all other users. To avoid this problem, a system is used whereby K_i can be feasibly computed from K_j if and only if $U_i \leq U_j$.

The keys K_i are generated as follows. A public integer t_i is assigned to each class U_i with the property

$$t_j \mid t_i \qquad \text{if and only if } U_i \leq U_j. \tag{1}$$

The authority U_0 chooses a random secret key K_0 and a secret pair of large prime numbers p and q, whose product $M = pq$ is made public. Then

$$K_i = K_0^{t_i} (\text{mod } M)$$

is communicated to U_i. If $U_i \leq U_j$, then t_i/t_j is an integer by (1), and U_j can compute K_i by the formula

$$K_i = K_0^{t_i} = K_0^{t_j(t_i/t_j)} = K_j^{t_i/t_j} (\text{mod } M).$$

However, if $U_i \not\leq U_j$, then t_i/t_j is not an integer and this computation is considered infeasible. This is discussed in [1] and relies on the fundamental assumption behind the RSA public key scheme: that it is difficult to extract roots modulo M, if M is the product of two unknown primes.

The only remaining question is how to choose the integers t_i. Fig. 1 shows the Hasse diagram of a poset where the t_i associated with class U_i is indicated inside the node representing that class. This assignment clearly satisfies condition (1), namely, that $t_j \mid t_i$ if and only if $U_i \leq U_j$. Unfortunately,

Manuscript received November 11, 1983; revised May 30, 1984. This work was supported by the Natural Sciences and Engineering Research Council of Canada under Strategic Grant G0381.

S. J. MacKinnon is with the Thousand Island Secondary School, Brockville, Ont., Canada, and the Department of Mathematics and Statistics, Queen's University, Kingston, Ont., Canada.

P. D. Taylor is with the Department of Mathematics and Statistics, Queen's University, Kingston, Ont., Canada.

H. Meijer and S. G. Akl are with the Department of Computing and Information Science, Queen's University, Kingston, Ont., Canada.

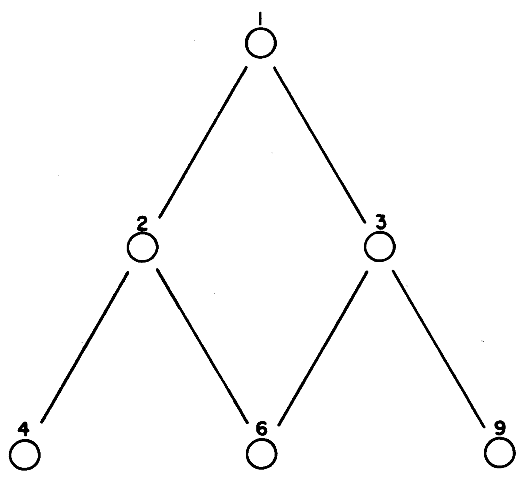

Fig. 1. An assignment of t_i to a poset.

such an ad hoc choice of t_i suffers from a serious weakness: two or more users belonging to different classes may be able to successfully cooperate to discover a key to which they are not entitled. Typically, in the example of Fig. 1, two users from U_i and U_j with $K_i = K_0^4$ and $K_j = K_0^9$ can easily find K_0 from the product

$$(K_i)^{-2}K_j = K_0^{-8}K_0^9 = K_0(\text{mod } M)$$

and hence compute all the keys in the system! It is proved in [1] that if a group of users who are not entitled to some key K_i manages to compute a product of integer powers of their keys and obtain $K_0^{t_i} = K_i(\text{mod } M)$, then t_i must be an integral combination of their t_j's. Since any such integral combination is a multiple of the gcd of these t_j's, this attack can be thwarted by ensuring that this gcd does not divide t_i.

In order to meet this new condition, it is suggested in [1] that the t_i's be computed from

$$t_i = \prod_{U_j \not\leq U_i} p_j \tag{2}$$

where $\{p_j\}$ is a sequence of distinct primes chosen by U_0. It is easy to show that such an assignment satisfies condition (1). Furthermore, collaborative attacks are not possible since the fact that $p_i \nmid t_i$ implies

$$\gcd_{U_j \not\leq U_i} (t_j) \nmid t_i$$

and hence no group of users who are not entitled to it can collaborate to find K_i. Fig. 2 shows the same poset as in Fig. 1: underneath each node is the prime number associated with it, and inside it is the corresponding t_i value computed as in (2).

As the small example of Fig. 2 shows, however, the problem with assignment (2) is that the t_i's can get quite big even for a small number of classes, thus slowing down the key computations. To illustrate this point, assume that the primes p_i assigned are the N smallest primes. If each of these primes is assigned to a class, then any U_i with no subordinate will have the property

$$t_i = \prod_{j \neq i} p_j .$$

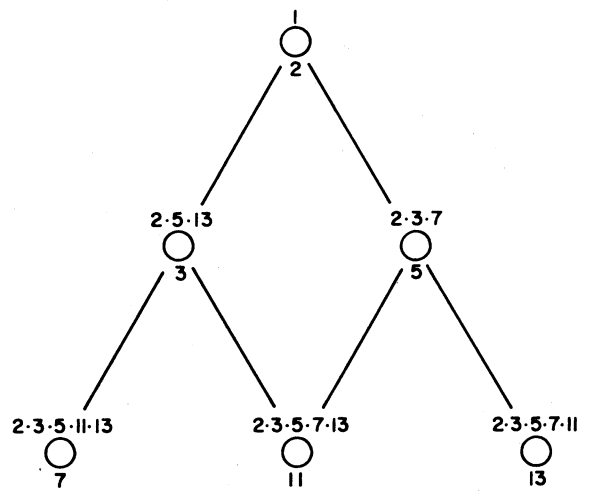

Fig. 2. An assignment with p_i shown below and t_i shown above node i.

In the worst case, $p_i = 2$, and t_i is equal to the product of the first $N - 1$ odd primes, which for $N = 20$ is already $\sim 10^{26}$. In general, the size of the Nth prime is $O(N \ln N)$, and hence t_i is $O((N \ln N)^N)$.

In the remainder of this paper we address the problem of finding t_i's whose size is smaller than those obtained from (2). For ease of notation we restrict attention to the set S giving it the partial order inherited from the U_i:

$$i \leq j \qquad \text{if and only if } U_i \leq U_j .$$

The problem can now be stated: given an arbitrary poset $\{S, \leq\}$ with a maximum element, find an assignment of integers $\{t_i : i \in S\}$ which in some sense is small and which satisfies

a) $t_j \mid t_i$ if and only if $i \leq j$,
b) $\gcd_{j \not\leq i} t_j \nmid t_i$.

III. The Canonical Assignment

It would at first appear that for any given poset there are many diverse sets of t_i's that satisfy properties a) and b). However, we shall show that any such set contains what we shall call a canonical set of t_i's which also exhibits the two necessary properties. Furthermore, any effort to keep the t_i's as small as possible will also lead to a canonical set.

A canonical set is defined in a manner similar to that of (2). A prime power n_i is first assigned to every node i, and the t_i's are computed as the lowest common multiple of the n_j from nodes not below node i. But unlike (2), where the n_i's were distinct primes, we will allow various powers of the same prime to be assigned so there will be cases where $n_j \mid n_i$. A well-designed mechanism for assigning the n_i's is needed to preserve property a).

The following algorithm produces a canonical assignment $\{t_i\}$ to nodes. The poset is first decomposed into disjoint chains. (A chain is a totally ordered subset.) Each chain is assigned a distinct prime. For each node i, we define

$$n_i = p^m$$

where i is the mth node from the top in the chain whose prime is p. Once all n_i's are thus determined, the t_i's are computed from the formula

$$t_i = \operatorname*{lcm}_{j \not\leq i} n_j .$$

The algorithm is illustrated in Fig. 3. Of course, for any given poset there will be many canonical assignments depending on the decomposition into chains and the assignment of primes to the chains.

We now prove two theorems. Theorem 1 shows that the above canonical construction satisfies a) and b) (it is easy to verify that (3) below is satisfied), and Theorem 2 shows that any assignment satisfying a) and b) "contains" a canonical assignment.

Theorem 1: Suppose S is a partially ordered set with an assignment of a prime power n_i to each i in S satisfying

$$n_i \mid n_j \Rightarrow i \geq j. \tag{3}$$

If $t_i = \operatorname{lcm}_{j \not\leq i} n_j$, then $\{t_i\}$ satisfies a) and b).

Proof: We first show $n_i \nmid t_i$. By assumption $n_i = p^m$ for some prime p, and t_i is the lcm of a set of numbers, none of which is divisible by p^m (by 3), so t_i cannot be divisible by p^m.

To show a) suppose $i \leq j$. Then $\{k \not\leq j\} \subseteq \{k \not\leq i\}$, and so

$$\operatorname*{lcm}_{k \not\leq j} n_k \mid \operatorname*{lcm}_{k \not\leq i} n_k ,$$

which means $t_j \mid t_i$. Conversely, if $i \not\leq j$, then $n_i \mid t_j$ (by definition of t_j), and hence $t_j \nmid t_i$, since $n_i \nmid t_i$.

Now we show $\{t_i\}$ satisfies b). If $i \not\leq j$, then $n_i \mid t_j$ (by definition of t_j), hence n_i divides the gcd of all such t_j. Since $n_i \nmid t_i$, we deduce b).

For the purposes of the next theorem let us call an assignment $\{t_i\}$ satisfying a) and b) *minimal* if whenever $\{s_i\}$ is another assignment satisfying a) and b) with $s_i \mid t_i$ for all i, then $s_i = t_i$. Clearly, any assignment $\{t_i\}$ satisfying a) and b) has a minimal such assignment $\{s_i\}$ with $s_i \mid t_i$.

Theorem 2: Any minimal assignment $\{t_i\}$ is canonical. That is, there is a decomposition of S into disjoint chains, and an assignment of distinct primes to these chains, so that for each node i, $t_i = \operatorname{lcm}_{j \not\leq i} n_j$ where we set $n_i = p^m$ when i is the mth (from the top) node in the chain whose prime is p.

Proof: Let $d_i = \gcd_{j \not\leq i} t_j$. By b), $d \nmid t_i$, so there is a prime p for which $p^m \nmid t_i$ where m is the number of times p occurs in the factorization of d_i. Let $n_i = p^m$ and $s_i = \operatorname{lcm}_{j \not\leq i} n_j$. We first show $s_i \mid t_i$. It is enough to show $n_j \mid t_i$ whenever $j \not\leq i$, and this follows since $n_j \mid d_j$ (by definition of n_j) and $d_j \mid t_i$ (by definition of d_j) when $i \not\geq j$. We now show $\{s_i\}$ satisfies a) and b). By Theorem 1 it is enough to show the n_i's satisfy (3). If $j \not\leq i$, then $n_j \mid s_i$, and hence $n_j \mid t_i$ (since $s_i \mid t_i$). It follows that $n_i \nmid n_j$, for otherwise $n_i \mid t_i$, a contradiction. Our assumption that $\{t_i\}$ is minimal now allows us to conclude that $s_i = t_i$.

It remains to show the existence of the decomposition. The subsets of the decomposition will be sets of nodes with a common prime. To show that any such subset is in fact totally ordered, suppose i and j are two nodes for which $n_i = p^m$ and $n_j = p^k$, with $m \leq k$. Then $n_i \mid n_j$, and (3), shown above to hold, implies $i \geq j$. So each subset is totally ordered with larger i having n_i with smaller powers of the prime. Now the canonical assignment with the same decomposition and set of primes has a set of n_i's which divide the current values, so the canonical t_i's also divide the current values, so our assumption of minimality allows us to conclude that $\{t_i\}$ is canonical.

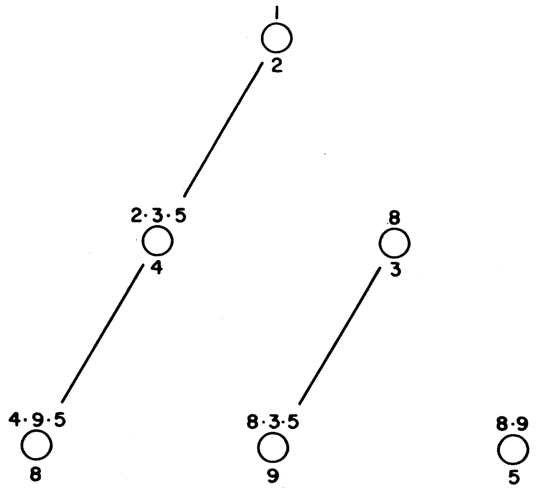

Fig. 3. One possible chain decomposition showing n_i below and t_i above node i.

IV. Optimization Issues

We now address the question of how, given a poset, an optimal canonical assignment might be obtained. What consitutes optimality will in part be determined by the uses we wish to make of the communication system (traffic patterns, etc.), and different objective functions will give rise to different canonical assignments. We remark that for almost any reasonable objective function, once we have a decomposition of the poset into chains, the optimal assignment will be determined by assigning the smallest primes to the longest chains. So our problem, for a given objective function, is one of finding the optimal decomposition. Exhaustive enumeration of all decompositions is an exponential process, however, and is clearly infeasible. We will, therefore, be interested in cases in which this problem can be shown to be equivalent to a known problem with a feasible algorithm, i.e., one whose running time is polynomial in $|S|$.

As a first example we consider the problem of minimizing the total number of primes used. This is the problem of finding a decomposition of a poset into a minimal number of chains, which was shown by Dantzig and Hoffman [2] to be equivalent to a linear programming problem of "transportation" type for which all basic feasible solutions are integral. Thus, Khachiyan's algorithm [6] will solve this problem in polynomial time. Alternatively, the problem can be formulated as a network flow problem [5] and can be solved with a flow-augmenting path algorithm requiring at most $O(|S|^3)$ steps [4]. It is known [5] that these representations also provide proofs of a theorem of Dilworth [3] that the number of chains in a minimal decomposition is equal to the maximum number of incomparable elements.

As a second objective function to be minimized, consider

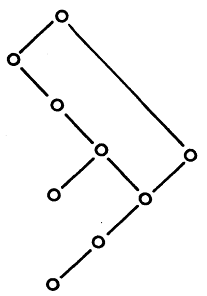
Fig. 4. Poset for Example 1.

$$[\operatorname{lcm}_{i} n_i]. \tag{4}$$

It is interesting to point out here that if a new node b is created with $b \leq i$ for all $i \in S$, then

$$t_b = \operatorname{lcm}_{i \not\leq b} n_i = \operatorname{lcm}_{i \in S} n_i,$$

and hence minimizing the objective function (4) is equivalent to minimizing the integer associated with the *least* element of the poset (if such an element exists). We also note that minimizing this objective function is not equivalent to minimizing the objective function discussed above, namely, the number of chains in a chain decomposition of the poset, as illustrated by the following example.

Example 1: Consider the poset in Fig. 4. There are two ways of decomposing this poset into a minimum number of chains as shown in Fig. 5. Both decompositions yield

$$\operatorname{lcm}_{i} n_i = 2^5 \cdot 3^4 = 2592.$$

However, the decomposition depicted in Fig. 6 (which does not minimize the number of chains) is better in terms of our second objective function as it yields

$$\operatorname{lcm}_{i} n_i = 2^7 \cdot 3 \cdot 5 = 1920.$$

From this example, it is clear that a special (polynomial-time) algorithm is needed to minimize our second objective function. The desirability of matching small primes with long chains suggests the following heuristic algorithm.

```
Algorithm: Longest Chain
    Step 1: Find the longest chain $\{i_1, \cdots, i_k\}$ in the poset.
    Step 2: Assign to this chain the smallest available prime $p$
            (which now becomes unavailable).
    Step 3: Remove nodes $i_1, \cdots, i_k$ from the poset.
    Step 4: If the poset is not empty, go to Step 1.
```

Although its running time is $O(|S|^2)$, it should be emphasized that this algorithm is just an heuristic. The example below shows the algorithm may fail to minimize either of the above objective functions.

Example 2: Consider the poset in Fig. 7. The longest chain algorithm will find the decomposition in three chains shown in Fig. 8, which yields

$$\operatorname{lcm}_{i} n_i = 2^5 \cdot 3 \cdot 5 = 480.$$

However, a better decomposition for both objective functions exists. This is shown in Fig. 9; it is composed of only two

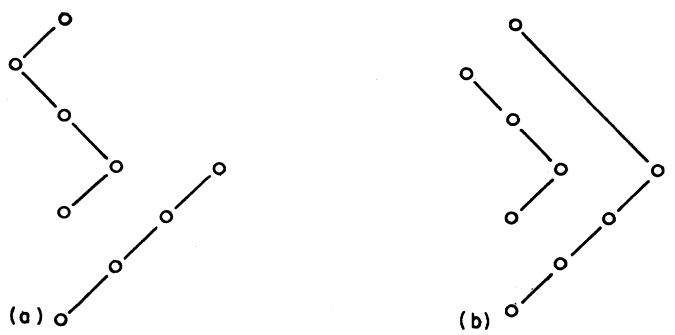

Fig. 5. Two ways of decomposing the poset in Fig. 4 into a minimum number of chains.

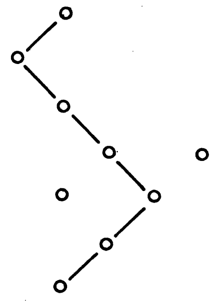
Fig. 6. A decomposition of the poset in Fig. 4 minimizing the second objective function.

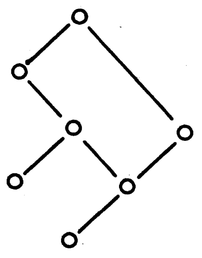
Fig. 7. Poset for Example 2.

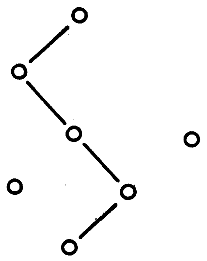
Fig. 8. Decomposing the poset of Fig. 7 into three chains by the longest chain algorithm.

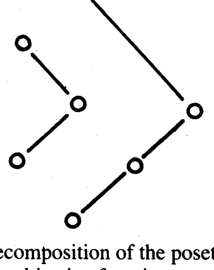
Fig. 9. A better decomposition of the poset of Fig. 7 for both objective functions.

chains and yields

$$\operatorname*{lcm}_i n_i = 2^4 \cdot 3^3 = 432 .$$

The existence of an algorithm for decomposing S into chains so as to minimize the objective function (4), and whose running time is a polynomial in $|S|$, remains an open problem.

Finally, we turn to an estimate and comparison of the size of the numbers arising in the assignment schemes of Sections II and III. First, we examine a canonical assignment. To obtain an estimate, we have taken a poset with a simple layered structure: there are L layers, each with k $(k > 1)$ times as many elements as in the layer above, and every element is $\leq$ all elements in any strictly higher layer. With one element in the top layer, the total number of elements is $N = (k^L - 1)/k - 1$. Using the longest chain algorithm, we get approximately (k may not be an integer) $k^{L-l} - k^{L-l-1}$ chains of length l for every l, $1 \leq l \leq L$. Using the smallest primes for the longest chains and assuming the nth prime is of size $n \ln(n)$, it is straightforward to find an expression for the size S_2 of the objective function (4), the lcm of all prime powers used. We get

$$S_2 \simeq \prod_{l=0}^{L-1} (k^l \ln k^l)^{(L-l)(k-1)k^{l-1}}. \tag{5}$$

An asymptotic estimate (which ignores the lnk^l in the base) gives $\log_k S_2 \sim N(L - (k + 1)/(k - 1))$, and since $L \simeq \log_k N(k - 1)$, we have

$$\ln S_2 \sim N\left[\ln N + \ln(k - 1) - \frac{k+1}{k-1} \ln k\right]. \tag{6}$$

Recall that, under the assignment of Section II, there was one prime used for every user, and the lcm of all numbers used was

$$S_1 \simeq \prod_{n=1}^{N} n \ln n , \tag{7}$$

which gives the asymptotic estimate

$$\ln S_1 \sim N \ln N . \tag{8}$$

So for both S_i it is the case that the number of decimal digits in the lcm *per user* (measured by $(\log_{10} S_i)/N$) grows like $\ln N$.

To get a better comparison for small k and $N \sim 100$, we have calculated values of $(\log_{10} S_i)/N$ using (5) and (7), and tabulated these in Tables I and II. It is seen that the number of digits in S_i per user grows linearly with $\ln N$, for both i (with k fixed for $i = 2$), but for N up to 200, the canonical assignment gives fewer digits for $k \leq 3$. For example, for $k = 2$ (each level twice the size of the next one up) and 127 users, the canonical assignment needs 64 primes, and S_2 has $(127)(1.66) = 210$ decimal digits. On the other hand, for $N = 127$, S_1 has about 285 digits. The canonical assignment is only a slight improvement. This is to be expected. For $k = 2$, fully half of all the chains used in our canonical assignment are of length one, each requiring a new prime, and of the remainder, half are of length 2, etc. Posets for which the layer sizes grow arithmetically rather than geometrically can be expected to require much smaller numbers under a canonical assignment.

TABLE I
ESTIMATE OF SIZE OF NUMBERS REQUIRED IN AN ASSIGNMENT OF A NEW PRIME TO EACH USER

# users N	# decimal digits in S_1 $\log_{10}S_1$	# digits per user
50	86	1.72
100	212	2.12
200	499	2.49

TABLE 2
ESTIMATE OF SIZE OF NUMBERS REQUIRED FOR A CANONICAL ASSIGNMENT USING THE LONGEST CHAIN ALGORITHM IN A POSET WITH A LAYERED STRUCTURE

	# Layers L	# Users N	# Primes Used	# decimal digits in S_2 $\log_{10}S_2$	# decimal digits per user
k=1.5	9	75	26	76	1.02
	10	113	38	141	1.24
	11	171	57	251	1.46
	12	257	86	435	1.69
k=2.0	6	63	32	80	1.28
	7	127	64	210	1.66
	8	255	128	519	2.04
k=2.5	5	64	39	95	1.48
	6	162	98	321	1.98
	7	406	244	1005	2.47
k=3	4	40	27	54	1.34
	5	121	81	235	1.95
	6	364	243	923	2.54

Finally, we mention a couple of open problems. It is not easy, with our schemes, to see how a new user could be accommodated without a key change throughout most of the system. Are there reasonable ways to handle this? Secondly, perhaps one can identify different types of posets, of which, for example, the layered structure of the last example would be one, and attempt to find optimal algorithms for each type.

ACKNOWLEDGMENT

The authors thank the referees for a number of useful suggestions.

REFERENCES

[1] S. G. Akl and P. D. Taylor, "Cryptographic solution to a problem of access control in a hierarchy," *ACM Trans. Comput. Syst.*, vol. 1, no. 3, Aug. 1983.

[2] G. B. Dantzig and A. J. Hoffman, "Dilworth's theorem on partially ordered sets," in *Linear Inequalities and Related Systems* (Annals of Mathematics Study 38). Princeton, NJ: Princeton Univ. Press, 1956, pp. 207–214.

[3] R.P. Dilworth, "A decomposition theorem for partially ordered sets," *Ann. Math.*, vol. 51, pp. 161–166, 1950.
[4] J. Edmonds and R.M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *J. ACM*, vol. 19, pp. 248–264, 1972.
[5] L.R. Ford and D.R. Fulkerson, *Flows in Networks*. Princeton, NJ: Princeton Univ. Press, 1962.
[6] L. Khachiyan, "A polynomial algorithm in linear programming," *Dokl. Akad. Nauk SSSR*, vol. 224, pp. 1093–1096, 1979. (Transl.: *Soviet Math. Dokl.*, vol. 20, pp. 191–194.)

Stephen J. MacKinnon received the B.A. degree in mathematics and the B.Ed., B.Sc., and M.Sc. degrees in 1975, 1976, 1978, and 1983, respectively, from Queen's University, Kingston, Ont., Canada, where his masters thesis was on recent developments in cryptology.

He has been a teacher of computer science and mathematics at Thousand Island Secondary School, Brockville, Ont., since 1976.

Peter D. Taylor received the Ph.D. degree in mathematics from Harvard University, Cambridge, MA, in 1969.

He is a Professor in the Department of Mathematics and Statistics at Queen's University, Kingston, Ont., Canada. His research interests are in population genetics, cryptology, and mathematics education.

Henk Meijer received the M.Sc. degree in econometrics from the Rijks-Universiteit Groningen, The Netherlands, in 1977 and the M.Sc. degree in computing science and the Ph.D. degree in mathematics from Queens University, Kingston, Ont., Canada, in 1979 and 1983, respectively.

He is an Assistant Professor in the Department of Computing and Information Science at Queen's University. His research interests are in the areas of computational complexity theory and cryptology.

Selim G. Akl (S'74–M'78–SM'85) received the B.Sc. and M.Sc. degrees in electrical engineering from the University of Alexandria, Alexandria, Egypt, in 1971 and 1975, respectively, and the Ph.D. degree in computer science from McGill University, Montreal, P.Q., Canada, in 1978.

He is currently an Associate Professor of Computing and Information Science at Queen's University, Kingston, Ont., Canada. His research interests are primarily in the area of algorithm design and analysis, in particular, for problems in computer security and parallel computation.

Dr. Akl is Editor of the *IACR Newsletter*, published by the International Association for Cryptologic Research, author of *Parallel Sorting Algorithms* (Academic, to be published), and coauthor of *The Convex Hull Problem* (Plenum, to be published). He is a founding member of the Canadian Applied Mathematics Society, and a member of the International Association for Cryptologic Research, the IEEE Computer Society, and the Association for Computing Machinery.