The Organization of Computations for Uniform Recurrence Equations

RICHARD M. KARP, RAYMOND E. MILLER, AND SHMUEL WINOGRAD

IBM Watson Research Center, Yorktown Heights, New York

ABSTRACT. A set of equations in the quantities $a_i(p)$, where $i = 1, 2, \cdots, m$ and p ranges over a set R of lattice points in n-space, is called a *system of uniform recurrence equations* if the following property holds: If p and q are in R and w is an integer n-vector, then $a_i(p)$ depends directly on $a_j(p - w)$ if and only if $a_i(q)$ depends directly on $a_j(q - w)$. Finite-difference approximations to systems of partial differential equations typically lead to such recurrence equations. The structure of such a system is specified by a *dependence graph* G having m vertices, in which the directed edges are labeled with integer n-vectors. For certain choices of the set R, necessary and sufficient conditions on G are given for the existence of a schedule to compute all the quantities $a_i(p)$ explicitly from their defining equations. Properties of such schedules, such as the degree to which computation can proceed "in parallel," are characterized. These characterizations depend on a certain iterative decomposition of a dependence graph into subgraphs. Analogous results concerning implicit schedules are also given.

1. *Introduction*

The process of setting up a problem for solution on a digital computer involves several interrelated considerations. It is necessary to choose an appropriate analytic form for the problem, construct an algorithm for its numerical solution, and organize this algorithm to make efficient use of the available computational facilities.

In the field of differential equations, a great deal of knowledge has been gained about the construction of difference equations whose solutions approximate the solution of a given differential equation. However, less attention has been given to the problems of scheduling and allocation involved in organizing the associated computations so as to make efficient use of the available storage and computing facilities. With the recent development of computers capable of performing many operations concurrently, these problems have taken on considerable importance.

In this paper some combinatorial aspects of computation processes are investigated which arise in the solution of partial differential equations by finite-difference methods. For this purpose, the properties of systems of uniform recurrence equations are introduced and studied. Most finite-difference schemes (for example, the usual point-relaxation methods for solving the Poisson equation) can be represented as uniform recurrence equations.

Properties are derived which characterize the number of computations that can be performed simultaneously in evaluating uniform recurrence equations, and an organization of the computations which gives efficient utilization of storage is described. In addition to aiding in the organization of computation processes, these properties provide insight into the desired features of the computing systems on which these computations are to be performed.

In Section 2 terminology is introduced. In Section 3 necessary and sufficient conditions are developed for a system of uniform recurrence equations to have an

Journal of the Association for Computing Machinery, Vol. 14, No. 3, July 1967, pp. 563–590.

explicit solution over certain regions. The special case of a single equation is considered in Section 4, and for this case a geometric interpretation of the necessary and sufficient conditions is given. Connections are established between the fastest possible schedule of the computation and optimum solutions to certain linear programs. It is shown (Theorem 5) that for a large class of equations, the amount of parallelism inherent in the computation is unbounded. In such problems, the number of operands required for subsequent operations is also shown to be unbounded (Theorem 6). This gives rise to the question of efficient utilization of a small high-speed memory in conjunction with a large bulk storage. A general scheme for the efficient utilization of such a storage configuration is described; an example of this scheme was previously considered [5].

In Section 5 some of the results of Section 4 are generalized to systems of equations. An effective test of the necessary and sufficient conditions of Section 3 is derived. The amount of parallelism inherent in the computations of a system of equations is also studied, and is shown to be qualitatively different than in the case of a single equation. Finally, in Section 6, related results are obtained for systems of equations which have an implicit, but no explicit, solution.

The special case in which all of the variables are scalars is investigated in greater detail in [6]. Also, quite apart from the application of uniform recurrence equations to finite-difference methods, the results of this paper may be applied to questions of the uniqueness of analytic solutions to systems of partial differential equations, using the approach initiated by Thomas [7].

2. *Terminology and Definitions*

Let L_n denote the lattice points in Euclidean n-space E_n. Any element p of L_n can be designated by an n-dimensional vector in which each coordinate is an integer. For each subset $R \subseteq L_n$ we let $\bar{R}$ denote $L_n - R$. We consider problems associated with the evaluation of a system of functions $a_1(p), a_2(p), \cdots, a_m(p)$ for all points $p \in R$. The values of $a_1(p), a_2(p), \cdots, a_m(p)$ are required to satisfy a system of m-recurrence equations having "uniform dependence," and boundary values are assumed to be given at points $p \in \bar{R}$ wherever required for function evaluation at points $p \in R$.

Equations having uniform dependence are best explained by first considering the case $m = 1$. The recurrence equation is of the form

$$a_1(p) = f_1(a_1(p - w_1), a_1(p - w_2), \cdots, a_1(p - w_s)),$$

where: $p \in R$,

w_j, $j = 1, 2, \cdots, s$, is an n-dimensional vector with integer coordinates, and

f_1 is a single-valued function which is strictly dependent on each of its s-variables.

A function $f(x_1, x_2, \cdots, x_m)$ is *strictly dependent* on x_i if, whatever values of b_j are assigned to the x_j, $j \neq i$, $f(b_1, \cdots, b_{i-1}, x_i, b_{i+1}, \cdots, b_m)$ is not a constant.

For any $p \in R$ the points $(p - w_1), (p - w_2), \cdots, (p - w_s)$ are elements of L_n. The vectors $w_1, w_2, \cdots, w_s$ are constants independent of p, and for this reason the equation is said to have *uniform dependence.*

Systems of m-equations with uniform dependence can be defined in terms of a structure called a dependence graph. A *dependence graph* G is a finite-directed graph with m vertices $v_1, v_2, \cdots, v_m$, a set of directed edges, and, associated with each edge e_i, an integer n-vector w_i.

Let $e_{i_1}, e_{i_2}, \cdots, e_{i_k}$ be the edges of G directed out of vertex v_i. Let these edges be directed into vertices $v_{i_1}, v_{i_2}, \cdots, v_{i_k}$, respectively; let w_{i_j} be the n-vector associated with e_{i_j}. The equation determined by v_i, then, is of the form

$$a_i(p) = f_i(a_{i_1}(p - w_{i_1}), a_{i_2}(p - w_{i_2}), \cdots, a_{i_k}(p - w_{i_k})).$$

The function f_i is assumed to be strictly dependent on each of its arguments, but its properties are otherwise left unspecified.

The dependence graph shown in Figure 1 yields the following system of equations:

$$a_1(p) = f_1(a_2(p - w_1), a_3(p - w_3)),$$

$$a_2(p) = f_2(a_1(p - w_2), a_2(p - w_6), a_3(p - w_4), a_3(p - w_7)),$$

$$a_3(p) = f_3(a_2(p - w_5)).$$

It is convenient to introduce some terminology for graphs. Let H be a directed graph with vertices $\{n_1, n_2, \cdots\}$ and edges $\{b_1, b_2, \cdots\}$. These sets may be either finite or infinite. A *path* is a sequence of edges $\pi = (b_{i_1}, b_{i_2}, \cdots)$ such that if b_{i_j} is directed into vertex n_k, then $b_{i_{j+1}}$ is directed out of n_k. If the sequence π is finite, having t edges, then we say that the path π has *length* t, and denote this by $l(\pi) = t$. If the sequence π is infinite, we say $l(\pi) = \infty$. A path whose length is finite is called a *cycle* if the vertex into which its last edge is directed is also the vertex from which the first edge is directed. If all the vertices out of which the edges in a cycle π are directed are distinct, then π is called a *simple cycle*.

We are interested in the precedence relations between evaluations of the functions $a_j(p)$ at various points $p \in R$. For this purpose we introduce the following terminology, where pairs (k, q) are taken from the set $\{1, 2, \cdots, m\} \times L_n$. The pair (k, p) is said to *depend directly* on (l, q), denoted by $(k, p) \xrightarrow{1} (l, q)$, if and only if $p \in R$ and G has an edge e_i directed from v_k to v_l such that $p - w_i = q$. Thus, $(k, p) \xrightarrow{1} (l, q)$ if and only if $a_l(q)$ is one of the arguments in the evaluation of $a_k(p)$. Also, *t-step dependence* is defined inductively for all positive integers t as follows: $(k, p) \xrightarrow{0} (l, q)$ if $k = l$, $p = q$; and $(k, p) \xrightarrow{t} (l, q)$ if there exists (h, r) such that $(k, p) \xrightarrow{t-1} (h, r)$ and $(h, r) \xrightarrow{1} (l, q)$.

Finally, $(k, p) \rightarrow (l, q)$ if $(k, p) \xrightarrow{t} (l, q)$ for some positive integer t. In any ordering of the computation, $a_l(q)$ must be evaluated before $a_k(p)$ if $(k, p) \rightarrow (l, q)$.

The ordering of the computation for a system of uniform recurrence equations defined over a region R can be described by a graph Γ (different from the dependence graph) having vertex set $(\{1, 2, \cdots, m\} \times R) \cup \{b\}$, where b is a special symbol corresponding to the set of points in $\bar{R} = L_n - R$. Γ has an edge from

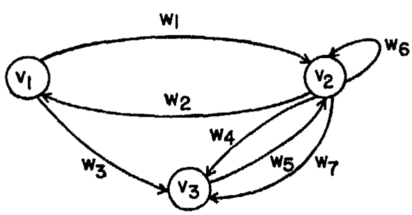


FIG. 1. A dependence graph

$(k, p) \in \{1, 2, \cdots, m\} \times R$ to $(l, q) \in \{1, 2, \cdots, m\} \times R$ if and only if $(k, p) \xrightarrow{1} (l, q)$, and an edge from (k, p) to b if and only if there exists (k', p') such that $p' \in \bar{R}$ and $(k, p) \xrightarrow{1} (k', p')$. Note that, although Γ has an infinite number of vertices if R is infinite, the number of edges directed out of any given vertex of Γ is finite. The relation $(k, p) \rightarrow (l, q)$ corresponds to the existence of a path from vertex (k, p) of Γ to vertex (l, q) of Γ. If $(k, p) \xrightarrow{t} (l, q)$, then there exists a path of length t in Γ directed from (k, p) to (l, q) if q is in R, and from (k, p) to b if q is in $\bar{R}$.

With these definitions we may introduce the notion of a schedule of computation. A *schedule* S is a function from $\{1, 2, \cdots, m\} \times R$ into the positive integers such that if $(k, p) \rightarrow (l, q)$ then $S(k, p) > S(l, q)$. The quantity $S(k, p)$ may be interpreted as the time at which $a_k(p)$ is computed. In this interpretation, the assumption is made that each function evaluation requires exactly one unit of time. This definition of a schedule ensures that, before $a_k(p)$ can be evaluated, all the arguments on which it depends have been computed. A schedule T is called *free* if:

$$T(k, p) = 1 \quad \text{if there is no } (l, q) \text{ such that } q \in R \text{ and } (k, p) \xrightarrow{1} (l, q);$$

$$T(k, p) = 1 + \max \{T(l, q) \mid q \in R \text{ and } (k, p) \xrightarrow{1} (l, q)\}, \quad \text{otherwise.}$$

The following facts are easily established. If there is a schedule, there is a unique free schedule. The free schedule, when it exists, is the "fastest" schedule possible (i.e., if S is a schedule, then $T(k, p) \leq S(k, p)$ for all (k, p)). The free schedule exists if and only if for each (k, p) such that $p \in R$ the function $F(k, p)$ exists, where

$$F(k, p) = \max \{t \geq 0 \mid \text{there exists } (x, q) \text{ such that } (k, p) \xrightarrow{t} (x, q)\}.$$

If the free schedule $T(k, p)$ exists, then it is given by

$$T(k, p) = F(k, p).$$

For convenience we define $T(k, p)$ even when a free schedule does not exist by setting

$$T(k, p) = \begin{cases} F(k, p) & \text{if} \quad F(k, p) \quad \text{is defined,} \\ \infty & \text{otherwise.} \end{cases}$$

The function $a_j(p)$ is *explicitly defined* if, for all $p \in R$, $T(j, p) < \infty$. Thus the existence of a schedule is equivalent to $a_j(p)$ being explicitly defined for $j = 1, 2, \cdots, m$.

The quantity $T(k, p)$ is clearly the length of a longest path of Γ directed out of the vertex (k, p), and $T(k, p) = \infty$ if and only if no such longest path exists.

Consider the following function associated with a schedule[1] S:

$$\phi_S(k, \tau) = \| \{p \mid S(k, p) = \tau\} \|, \qquad k = 1, 2, \cdots, m, \quad \tau = 1, 2, \cdots.$$

The schedule S has *bounded parallelism* if there exists an integer K such that $\phi_S(k, \tau) \leq K$ for all k and τ; otherwise, S has *unbounded parallelism*. Thus, unbounded parallelism means that either $\phi_S(k, \tau)$ is infinite for some (k, τ) or grows without bound for some k.

In the following sections we give necessary and sufficient conditions for a function $a_k(p)$ to be explicitly defined, and investigate the amount of parallelism for the free

[1] If U is a set, then $\| U \|$ denotes the cardinality of U.

schedule, as well as techniques for the efficient utilization of storage in computing explicitly defined functions.

3. *Criteria for a Function to Be Explicitly Defined*

The purpose of this section is to characterize systems of uniform recurrence equations which are explicitly defined throughout a region R. Necessary and sufficient conditions are given for a function $a_k(p)$ to be explicitly defined throughout the region F_n consisting of those n-dimensional vectors all of whose coordinates are positive integers. These conditions are also extended to the case in which R is the Cartesian product of F_n with a finite-point set.

In order to prove the main theorem of this section, we require the following well-known lemma about infinite graphs which gives an alternate characterization of the case $T(k, p) = \infty$.

LEMMA 1 (Berge) [1, p. 17, Cor. 1]. *Let H be a directed graph in which the number of edges directed out of each vertex is finite. Then, if there is no upper bound on the lengths of paths directed out of vertex v, there exists an infinite path directed out of v.*

Applying the lemma to Γ, we find that $T(k, p) = \infty$ if and only if Γ has an infinite path directed out of (k, p); i.e., $T(k, p) = \infty$ if and only if there is an infinite sequence $(k_1, p_1), (k_2, p_2), \cdots$ of elements of $\{1, 2, \cdots, m\} \times R$ such that $(k, p) = (k_1, p_1)$ and, for all i, $(k_i, p_i) \xrightarrow{1} (k_{i+1}, p_{i+1})$.

To state the theorem we introduce the following terminology. Let $\pi = (e_{i_1}, e_{i_2}, \cdots, e_{i_u})$ be a path in the dependence graph G. The *weight* $w(\pi)$ of π is defined by

$$w(\pi) = \sum_{j=1}^{u} w_{i_j} .$$

A vector is called *nonpositive* if all of its coordinates are nonpositive. A path π is called *nonpositive* if $w(\pi)$ is a nonpositive vector.

THEOREM 1. *Let G be a dependence graph specifying a system of uniform recurrence equations defined over the region F_n. Then the function $a_k(p)$ is explicitly defined if and only if G does not have a path from v_k to any vertex v_l contained in a nonpositive cycle.*

PROOF. Suppose there is a path π from v_k to v_l and a nonpositive cycle C including v_l, as shown in Figure 2. Let

$$X^{(j)} = \sum_{i=1}^{j} w^{(i)}.$$

Choose $X \in F_n$ such that $X - X^{(j)} \in F_n$, $j = 1, 2, \cdots, r + t$. Also, let $Y = X^{(r+t)}$

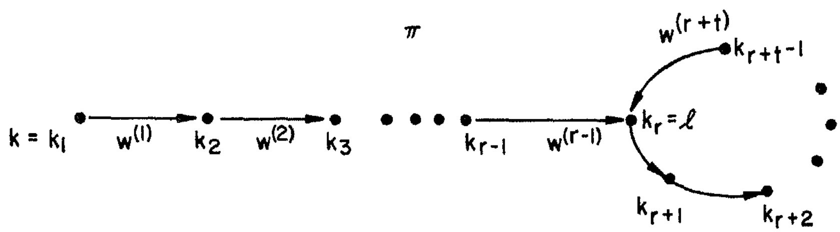


FIG. 2

[Correction: The term $k_{r+t} - 1$ at the upper right should read: k_{r+t-1} .—Ed.]

$- X^{(r-1)}$ (Y is the weight of the cycle C). Since C is nonpositive,

$$X - X^{(j)} - \alpha Y \in F_n, \qquad j = 1, 2, \cdots, r + t, \quad \alpha = 0, 1, 2, \cdots.$$

Thus

$$\begin{aligned}(k_1, X) &\xrightarrow{1} (k_2, X - X^{(1)}) \xrightarrow{1} \cdots \xrightarrow{1} (k_r, X - X^{(r-1)})\\ &\xrightarrow{1} (k_{r+1}, X - X^{(r)}) \xrightarrow{1} \cdots \xrightarrow{1} (k_r, X - X^{(r+t)}) \xrightarrow{1} (k_{r+1}, X - X^{(r)} - Y)\\ &\xrightarrow{1} \cdots \xrightarrow{1} (k_{r+j+1}, X - X^{(r+j)} - \alpha Y) \xrightarrow{1} \cdots,\end{aligned}$$

and therefore $T(k, X) = \infty$.

To complete the proof, we show that, if $T(k, p) = \infty$, then G has a path from v_k to a nonpositive cycle. By the remarks following Lemma 1, $T(k, p) = \infty$ implies the existence of an infinite sequence $\sigma = (k_1, p_1), (k_2, p_2), \cdots$ such that $(k, p) = (k_1, p_1)$ and, for all i, $(k_i, p_i) \xrightarrow{1} (k_{i+1}, p_{i+1})$.

From σ we may select an infinite subsequence $\{(k_i, p_i)\}$ nondecreasing in each coordinate of $\{p_i\}$. Such a sequence may be selected recursively by choosing a sequence nondecreasing in the first coordinate, then from this choosing a subsequence nondecreasing on the second coordinate, and so on. Since the indices k_i belong to a finite set, some value k must recur. Hence, the sequence contains a cycle $(k, p) \rightarrow (k, p - w)$, where w is nonpositive.

The method of proof of Theorem 1 may be extended to yield analogous results for certain other regions. In particular, we consider "strips"; i.e., R is of the form $R = Q_t \times F_{n-t}$, where Q_t is a finite subset of F_t.

COROLLARY 1. *Let G be a dependence graph specifying a system of uniform recurrence equations defined over the region $R = Q_t \times F_{n-t}$. Then $a_k(p)$ is explicitly defined if and only if there do not exist $l \in \{1, 2, \cdots, m\}$ and $p, q, r \in R$ such that:*

(i) $(k, p) \rightarrow (l, q) \rightarrow (l, r)$;

(ii) *the vector $q - r$ is zero in the coordinates corresponding to Q_t, and nonpositive in the coordinates corresponding to F_{n-t}.*

PROOF. Consider the dependence graph G' with vertex set $\{1, 2, \cdots, m\} \times Q_t$, and edges as follows: There is an edge of G' from (k, q_1) to (l, q_2) labeled u if and only if there is an edge of G from k to l labeled $w = (w_q, w_f)$, where $w_q = q_1 - q_2$ and $w_f = u$. It is clear that $T(k, (q, p))$ with respect to the system of equations over R specified by G is equal to $T((k, q), p)$ with respect to the system of equations over F_{n-t} specified by G'. The corollary thus follows from Theorem 1.

Corollary 1 has the disadvantage that its statement depends on the relation "$\rightarrow$," which is a property of the region considered as well as the dependence graph G. The following result, which is implied by Corollary 1, involves only G.

COROLLARY 2. *Let G be a dependence graph specifying a system of uniform recurrence equations. Then the following are equivalent:*

(i) *for every finite subset $Q_t \subseteq F_t$, $a_k(p)$ is explicitly defined over the region $Q_t \times F_{n-t}$;*

(ii) *there is no vertex v_l such that* (a) *G has a path from v_k to v_l, and* (b) *v_l is contained in a cycle C whose weight is zero in the first t coordinates and nonpositive in the last $n - t$ coordinates.*

4. *Theorems for One Recurrence Equation*

In this section we consider the case of a single recurrence equation ($m = 1$), which takes on the form

$$a(p) = f(a(p - w_1), a(p - w_2), \cdots, a(p - w_s)).$$

We begin by giving several necessary and sufficient conditions for $a(p)$ to be explicitly defined over F_n. One of these conditions is that a certain system of linear inequalities has a feasible solution. The optimal solution of an associated linear program provides an upper bound on the free schedule $T(p)$, and this bound is shown to be tight for points sufficiently far from the boundary. From this bound, it is deduced that the free schedule has unbounded parallelism when $n \geq 2$. Finally, we present a class of schedules which have a specified upper bound on parallelism and make efficient use of temporary storage.

CONDITIONS FOR $a(p)$ TO BE EXPLICITLY DEFINED. A vector is called *semipositive* if all of its components are nonnegative and not all of them are zero.

THEOREM 2. *Consider the uniform recurrence equation* $a(p) = f(a(p - w_1), a(p - w_2), \cdots, a(p - w_s))$, *where* p *is an* n*-vector. The following statements are equivalent:*

(i) $a(p)$ *is explicitly defined over* F_n;

(ii) *there exists no semipositive vector* $(u_1, u_2, \cdots, u_s)$ *such that*

$$\sum_{j=1}^{s} -u_j w_j \geq 0;$$

(iii) *the system of inequalities*

$$w_j \cdot x \geq 1, \qquad j = 1, 2, \cdots, s,$$
$$x_i \geq 0, \qquad i = 1, 2, \cdots, n,$$

has a solution;

(iv) *for every* $p \in F_n$, *the following two linear programs have a common optimal value* $m(p)$:

$$\mathrm{I}\ \begin{cases} p - \sum_{j=1}^{s} u_j w_j \geq 0, \\ u_j \geq 0, \qquad j = 1, 2, \cdots, s, \\ \max \sum_{i=1}^{s} u_j . \end{cases} \qquad \mathrm{II}\ \begin{cases} w_j \cdot x \geq 1, \qquad j = 1, 2, \cdots, s, \\ x_i \geq 0, \qquad i = 1, 2, \cdots, w, \\ \min p \cdot x. \end{cases}$$

PROOF. Since the system of inequalities in (ii) is homogeneous and the vectors w_j are rational, the system has a semipositive solution if and only if it has a semipositive integer solution. But since the dependence graph in this case consists of a single vertex with loops labeled $w_1, w_2, \cdots, w_s$, the set of cycle weights is exactly the set of semipositive integer solutions to the given system. Thus, by Theorem 1, (i) and (ii) are equivalent. The equivalence of (ii) and (iii) follows from the Minkowski-Farkas lemma ([2, p. 137]), which states that for any $m \times n$ matrix A and n-vector d, exactly one of the following two statements holds:

(a) the system $Ax \leq d$ has a nonnegative solution;
(b) the system $u^T A \leq 0, \quad u^T d < 0$ has a nonnegative solution.
Setting

$$A = \begin{pmatrix} -w_1 \\ -w_2 \\ \vdots \\ -w_s \end{pmatrix} \quad \text{and} \quad d = \begin{pmatrix} -1 \\ -1 \\ \vdots \\ -1 \end{pmatrix},$$

this lemma yields the desired equivalence. Clearly, (iv) implies (iii), since the existence of an optimal value for program II implies that (iii) has a feasible solution. Conversely, (iii) implies that program II has a feasible solution; also, $u_j = 0$, $j = 1, 2, \cdots, s$ is a feasible solution of program I. But, from the duality theory of linear programming ([2, p. 129]), dual programs which are both feasible must have a common optimal value. Thus, (iv) follows from (iii).

Condition (iii) may be interpreted geometrically as follows: There is a hyperplane through the origin which separates the first orthant (except for the origin) from the vectors $-w_1, -w_2, \cdots, -w_s$. We call such a hyperplane, and by extension its normal vector, a *separating hyperplane.*

Example 1. Figure 3(a) represents a recurrence equation which is explicitly defined, and Figure 3(b) represents one which is not explicitly defined.

COROLLARY 3. *Consider the uniform recurrence equation* $a(p) = f(a(p - w_1), a(p - w_2), \cdots, a(p - w_s))$, *where* p *is an n-vector. The following statements are equivalent:*

(i) $a(p)$ *is explicitly defined for all regions* $R = Q_t \times F_{n-t}$ *in which* Q_t *is a finite subset of* F_t;

(ii) *there exists no semipositive vector* $(u_1, u_2, \cdots, u_s)$ *such that*

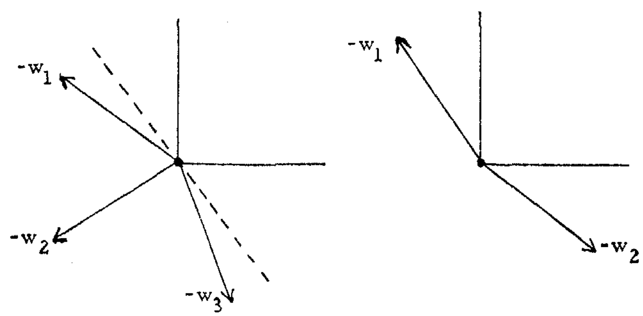
$$\sum_{j=1}^{s} -u_j w_j$$

is zero in the first t coordinates and nonnegative in the last $n - t$ *coordinates;*

(iii) *the system of inequalities*

$$\begin{aligned} w_j \cdot x \geq 1, &\qquad j = 1, 2, \cdots, s, \\ x_i \geq 0, &\qquad i = t+1, \quad t+2, \cdots, n, \end{aligned}$$

has a feasible solution.



(a) (b)

FIG. 3

RELATIONSHIP BETWEEN $m(p)$ AND $T(p)$. Consider two points $p, q \in R$ such that $p \xrightarrow{t} q$. Then there is an s-tuple of nonnegative integers $(u_1, u_2, \cdots, u_s)$ such that

$$q = p - \sum_{j=1}^{s} u_j w_j$$

and

$$\sum_{j=1}^{s} u_j = t.$$

If $R = F_n$ then $q > 0$ and $(u_1, u_2, \cdots, u_s)$ is a feasible solution to linear program I, whose optimal value is $m(p)$; hence $t \leq m(p)$. From the definition of the free schedule $T(p)$ it is immediate that $T(p) - 1 \leq m(p)$. We next investigate the relationship between $m(p)$ and $T(p)$ more closely. We show that a uniform upper bound on $m(p) - T(p)$ throughout F_n does not, in general, exist, but that such a bound does exist for all points $p \in F_n$ which are not "too close" to the boundary of the first orthant. What is meant by "too close" is made precise later.

The following example shows the impossibility of giving a bound on $m(p) - T(p)$ in general.

Example 2. Let $n = s = 3$,

$$w_1 = (-1, 1, 1), \qquad w_2 = (1, -1, 1), \qquad w_3 = (0, 0, 10).$$

For all points p of the form

$$p = (1, 1, 10j), \qquad j = 1, 2, 3, \cdots,$$

it is easily verified that

$$m(p) = 10j, \qquad T(p) = j.$$

Therefore no uniform upper bound for $m(p) - T(p)$ exists.

Now, to define a region for which a uniform bound on $m(p) - T(p)$ exists, let

$$\begin{cases} C_h = 1 \quad \text{if} \quad (w_j)_h = 0 \quad \text{for} \quad j = 1, 2, \cdots, s; \\ C_h = \sum_{j=1}^{s} |(w_j)_h| \quad \text{otherwise}; \quad h = 1, 2, \cdots, n. \end{cases}$$

Let U be the set of all n-vectors x such that

$$x_h \geq C_h, \qquad h = 1, 2, \cdots, n,$$

where the x_h are not necessarily integers. The points of $U \cap F_n$ are those points of F_n which are not "too close" to the boundary of the first orthant, as is made clear by the following theorem.

THEOREM 3. *There exists a constant K such that, for all points $p \in U \cap F_n$, $m(p) - T(p) < K$.*

The proof of this theorem is based on the fact that if $p = q + \Sigma n_i w_i$, $\Sigma n_i = N$, then there is a sequence

$$p, p - w_{i_1}, p - (w_{i_1} + w_{i_2}), \cdots, p - (w_{i_1} + \cdots + w_{i_N}) = q$$

all of whose points lie within a "bounded tube" around the line segment connecting p and q. The proof makes use of two lemmas.

LEMMA 2. *Let p and q be in F_n, such that:*
(i) $p = y + \sum_{i=1}^{s} \theta_i w_i$, $y \in U$, $0 \le \theta_i < 1$;
(ii) $p - \sum_{i=1}^{s} \alpha_i w_i = q$, $\alpha_i \in \{0, 1\}$.
Then $p \xrightarrow{N} q$, where $N = \sum_{i=1}^{s} \alpha_i$.

PROOF. With no loss of generality, let $\alpha_i = 1$ for $i = 1, 2, \cdots, l$ and $\alpha_i = 0$ otherwise. Consider the points $p = t_0, t_1, \cdots, t_l = q$, where

$$t_j = p - \sum_{i=1}^{j} w_i .$$

We show that each t_j is an element of F_n and thus obtain the desired path $p = t_0 \xrightarrow{1} t_1 \xrightarrow{1} \cdots \xrightarrow{1} t_l = q$. Consider

$$(t_j)_h = (p - \sum_{i=1}^{j} w_i)_h .$$

If $(w_i)_h \ge 0$ for $i = j + 1, j + 2, \cdots, j + l$, then

$$(t_j)_h \ge (p - \sum_{i=1}^{l} w_i)_h = (q)_h > 0.$$

Otherwise

$$(t_j)_h = \left(p - \sum_{i=1}^{j} w_i\right)_h = (y)_h + \sum_{i=1}^{j} (\theta_i - 1)(w_i)_h + \sum_{i=j+1}^{s} \theta_i (w_i)_h .$$

In this case, at least one of the $(w_i)_h$, $i = j + 1, j + 2, \cdots, s$, is not zero, and since $0 \le \theta_i < 1$, $-1 \le \theta_i - 1 < 0$. We obtain

$$(t_j)_h > (y)_h - \sum_{i=1}^{s} |(w_i)_h| = (y)_h - C_h \ge 0.$$

Thus $t_j \in F_n$, giving the desired result.

LEMMA 3. *Let p and q be in $U \cap F_n$ such that*

$$q = p - \sum_{i=1}^{s} \alpha_i w_i ,$$

where each α_i is a nonnegative integer; then $p \xrightarrow{N} q$, where

$$N = \sum_{i=1}^{s} \alpha_i .$$

PROOF. Let $\lfloor a \rfloor$ denote the integer part of the number a. Let

$$r_c = p - \sum_{i=1}^{s} \lfloor \frac{c}{N} \alpha_i \rfloor w_i = p - \frac{c}{N} \sum_{i=1}^{s} \alpha_i w_i + \sum_{i=1}^{s} \left(\frac{c}{N} \alpha_i - \lfloor \frac{c}{N} \alpha_i \rfloor\right) w_i ,$$
$$c = 0, 1, \cdots, N.$$

Note that $\lfloor [(c + 1)/N] \alpha_i \rfloor - \lfloor (c/N) \alpha_i \rfloor \in \{0, 1\}$ for all i, $c = 0, 1, \cdots, N$. Then applying Lemma 2 by letting r_c play the role of p, r_{c+1} the role of q, and $[(c/N)\alpha_i - \lfloor (c/N) \alpha_i \rfloor]$ the role of θ_i, we find that $r_c \xrightarrow{L} r_{c+1}$, where

$$L = \sum_{i=1}^{s} \left(\lfloor \frac{c+1}{N} \alpha_i \rfloor - \lfloor \frac{c}{N} \alpha_i \rfloor\right).$$

Joining these chains, we obtain the desired path $p \xrightarrow{N} q$.

Proof of Theorem 3. Let $m'(p)$ be the optimal solution of linear program I with the added restriction that the u_j are integers. Then it follows from the results of Gomory [3] that there exists a number K_1 such that $m(p) - m'(p) < K_1$ for all p.

Let π be the vector $\pi = (C_1, C_2, \cdots, C_n)$ and, for a given $p \in F_n$, let $(u_1, u_2, \cdots, u_s)$ be the s-tuple of nonnegative integers which maximizes $m'(p - \pi)$. Since

$$p - \pi - \sum_{i=1}^{s} u_i w_i \geq 0,$$

we have

$$p - \sum_{i=1}^{s} u_i w_i \geq \pi,$$

and by Lemma 3,

$$p \xrightarrow{m'(p-\pi)} p - \sum_{i=1}^{s} u_i w_i,$$

and therefore

$$T(p) \geq m'(p - \pi).$$

Therefore

$$m(p) - T(p) \leq m(p) - m'(p - \pi) = m(p) - m(p - \pi)$$
$$+ m(p - \pi) - m'(p - \pi) < m(p) - m(p - \pi) + K_1.$$

We show that $m(p) - m(p - \pi) \leq K_2$ for all p and thereby, letting $K = K_1 + K_2$, we prove the theorem.

Since the dual linear programs I and II both have optimal solutions,

$$m(p - \pi) = (p - \pi)\cdot x, \qquad m(p) = p\cdot y,$$

where x and y are optimal solutions of II for the points $(p - \pi)$ and p, respectively. Also,

$$p\cdot y \leq p\cdot x.$$

Therefore

$$m(p) - m(p - \pi) = p\cdot y - (p - \pi)\cdot x \leq p\cdot x - (p - \pi)\cdot x = \pi\cdot x \leq \max_z \pi\cdot z$$

where z ranges over the finite set of extreme points of the convex set of solutions of II. Thus we can let $K_2 = \max_z \pi\cdot z$, and the theorem is proved.

The problem of finding the minimal pairs (π, K) such that $m(p) - T(p) < K$ for all $p \geq \pi$, is unsolved. In general, the π and K resulting from Theorem 3 are not minimum. For the special case of two dimensions $(n = 2)$, it can be shown that π can be taken as $(0, 0)$; that is, there exists a K such that $m(p) - T(p) < K$ for all $p \in F_2$.

Amount of Parallelism. We now turn our attention to the amount of parallelism that arises in the computation of an explicitly defined function $a(p)$. Recalling our notation of Section 2, for a schedule S the parallelism is denoted by $\phi_S(\tau)$. The amount of parallelism for an explicitly defined function $a(p)$ is characterized by the following theorem.

THEOREM 4. *Let $a(p)$ be explicitly defined over F_n with $n \geq 2$. There exists a schedule S such that $\sup_\tau \phi_S(\tau) = \infty$.*

PROOF. Since $a(p)$ is explicitly defined there exists a nonnegative x such that $w_i \cdot x \geq 1$ for $i = 1, 2, \cdots, s$. The schedule S is defined as follows. At time τ all the points p for which $\tau - 1 < p \cdot x \leq \tau$ are computed. That S is a schedule is easily verified. It follows that

$$\sum_{\tau=1}^{t} \phi_S(\tau) = \|\{p \mid p \cdot x \leq t\}\| \geq \|\{p \mid |p| \leq t/|x|\}\|,$$

and the latter is not bounded by any linear function of τ when $n \geq 2$: hence $\phi_S(\tau)$ is not bounded.

More detailed properties of the free schedule T may be inferred from the separating hyperplanes x which exist as feasible solutions to linear program II. If there exists a separating hyperplane x having some component equal to zero, then $\phi_T(\tau) = \infty$ for all τ unless the problem is degenerate in the sense that the free schedule evaluates all the points $p \in F_n$ at $\tau = 1$. If there exists no separating hyperplane x having a component equal to zero, then $\phi_T(1)$ may either be finite or infinite. If $\phi_T(1) < \infty$, this implies that $\phi_T(\tau) < \infty$ for all τ. From Theorem 3, of course, it follows that $\|\{p \mid p \in U \cap F_n \text{ and } T(p) = \tau\}\|$ is finite. Nevertheless, $\phi_T(1)$ may be infinite if an infinite number of points in $F_n - U$ are computed in the free schedule at time $\tau = 1$. This computation of an infinite number of points in the region $F_n - U$ may either continue for each time τ or terminate such that $\phi_T(\tau) < \infty$ for all sufficiently large τ. These two possibilities are illustrated by the following two examples.

Example 3. Let $w_1 = (-3, 2, 2)$, $w_2 = (2, -3, 2)$, and $w_3 = (2, 2, -3)$. Then $x = (1, 1, 1)$ is a feasible solution to linear program II, and II has no feasible solution with zero components. Nevertheless, for every point p of the form $(2, 2, N)$, $(2, N, 2)$, and $(N, 2, 2)$, $T(p) = 1$. Therefore, $\phi_T(1) = \infty$. In this example $\phi_T(\tau) < \infty$ for all $\tau \geq 2$.

Example 4. Let $w_1 = (-2, 1, 1, 1)$, $w_2 = (1, -2, 1, 1)$, $w_3 = (1, 1, -2, 1)$, $w_4 = (1, 1, 1, -2)$, and $w_5 = (0, 0, 1, 0)$. Then $x = (1, 1, 1, 1)$ is a separating hyperplane and no separating hyperplane exists having a zero component. Here $T(p) = k$ for points p of the form $p = (1, 1, k, N)$ where $N = 1, 2, 3, \cdots$ so that $\phi_T(\tau) = \infty$ for all τ.

STORAGE REQUIREMENTS. Let S be a schedule for an explicitly defined function $a(p)$. We define $\sigma_S(\tau)$ as follows:

$$\sigma_S(\tau) = \| \{q \in F_n \mid S(q) \leq \tau \text{ and there exists } p \text{ such that } p \xrightarrow{1} q,\ S(p) > \tau\} \|.$$

The function σ_S measures the number of computed values which have to be retained at time τ in order to compute other points. Therefore, $\sigma_S(\tau)$ is a measure of the amount of storage required at time τ to compute $a(p)$ using schedule S.

THEOREM 5. *Suppose that $a(p)$ is explicitly defined over F_n. If there exist two vectors w_i and w_j which are linearly independent, then $\sup_\tau \sigma_S(\tau) = \infty$, for all schedules S.*

PROOF. We show that for any arbitrary schedule S and positive integer m, there exists a τ such that $\sigma_S(\tau) \geq m$. Let $p \in F_n$ such that the set $\{p + aw_i + bw_j \mid a, b = 0, 1, \cdots, m\}$ is a subset of F_n. Since w_i and w_j are independent, this set contains $(m+1)^2$ elements. Let $\tau = S(p + mw_j)$. Two cases arise.

Case I. $S(p + mw_i) > \tau$. In this case $S(p + mw_i + bw_j) > \tau$, $b = 0, 1, \cdots, m$. Let a_b be the largest integer a such that $S(p + a_b w_i + bw_j) \leq \tau$; clearly $0 \leq a_b < m$. For each b, however, $S(p + (a_b + 1)w_i + bw_j) > \tau$ and $p + (a_b + 1)w_i + bw_j \xrightarrow{1} p + a_b w_i + bw_j$. Therefore,

$$\sigma_S(\tau) \geq \| \{p + a_b w_i + bw_j\} \| = m + 1.$$

Case II. $S(p + mw_i) \leq \tau$. In this case, since

$$S(p + mw_j) = \tau, \quad S(p + aw_i + mw_j) > \tau$$

for $a = 1, 2, \cdots, m$. Let b_a be the largest integer b such that $S(p + aw_i + b_a w_j) \leq \tau$; clearly $0 \leq b_a < m$. For each a, however, $S(p + aw_i + (b_a + 1)w_j) > \tau$ and $p + aw_i + (b_a + 1)w_j \xrightarrow{1} p + aw_i + b_a w_j$. Therefore, $\sigma_S(\tau) \geq \| \{p + aw_i + b_a w_j\} \| = m$.

When the vectors w_i are collinear, and all their coordinates are nonnegative, then, for $n \geq 2$, the points of F_n lie on infinitely many distinct lines parallel to w_1, and each such line contains an infinite number of points in F_n. Once computation on such a line has begun, at least one computed point on that line must be retained at all times. Since any schedule S must eventually start on each line, $\sigma_S(\tau)$ must be unbounded.

If all the vectors are collinear and have at least one negative coordinate, then the lines parallel to the vectors are of finite length and a schedule S that computes all the points on one line before another line is started is a schedule for which $\sigma_S(\tau)$ is bounded for all τ. Clearly, in this case, schedules with unbounded $\sigma_S(\tau)$ also exist.

Theorem 5 shows that, except for special cases, the amount of storage required to compute the function $a(p)$ is unbounded. Suppose the computation is performed on a computer which has a limited-size high-speed storage, and an unbounded bulk storage of slower speed. The efficiency of a schedule for the computation must, for this case, take into account the time required for fetching operands from storage. For this reason it is advantageous to schedule the computation so that data in the high-speed storage is used as much as possible to compute new points. We describe a scheme which yields high utilization of the fast storage. As a simple illustration of the scheme consider the following example.

Example 5. Consider the two-dimensional recurrence equation

$$a(p) = f(a(p - w_1), a(p - w_2), a(p - w_3)),$$

where $w_1 = (3, -1)$; $w_2 = (-1, 3)$; $w_3 = (1, 1)$. Let $x_1 = (1, 2)$, $x_2 = (2, 1)$. It is easily verified that $w_j \cdot x_k \geq 1$, $j = 1, 2, 3$, $k = 1, 2$. Thus, by Theorem 2, $a(p)$ is explicitly defined over F_2.

To describe the computation scheme, assuming that five points can be computed simultaneously, assume that certain points of F_2 have been computed. More specifically, assume that we have computed all points p such that $p \cdot x_1 < 60$ or $p \cdot x_1 < 74$ and $p \cdot x_2 < 60$; in this case, the five points $p_1 = (20, 20)$, $p_2 = (19, 22)$, $p_3 = (18, 24)$, $p_4 = (17, 26)$, $p_5 = (16, 28)$ can be computed. This is illustrated in Figure 4. The crosshatched region covers the points assumed to have been com-

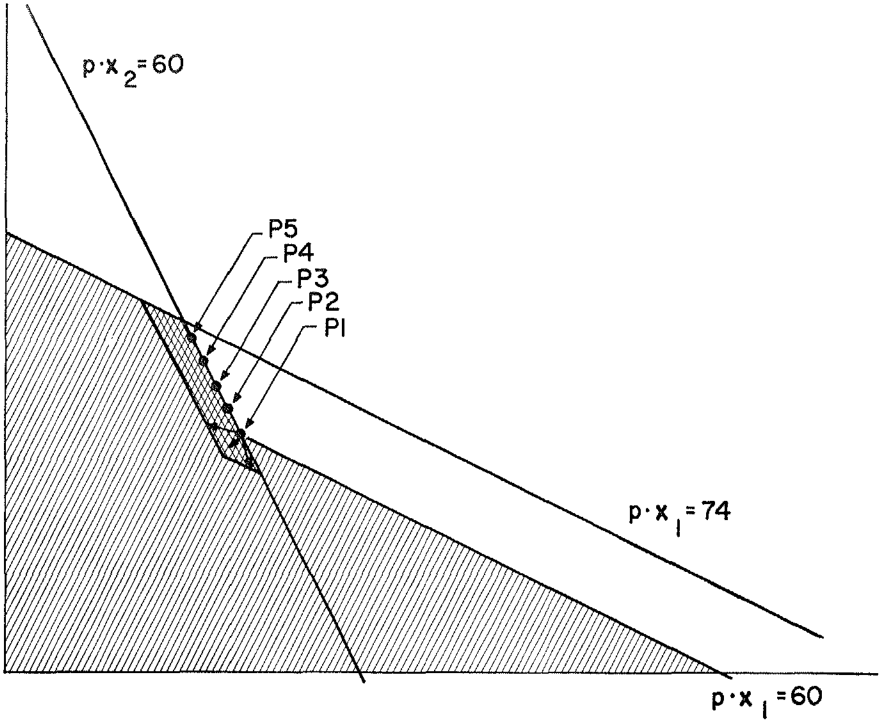


FIG. 4

puted. The double-crosshatched area indicates the region from which the operands required to compute p_1, p_2, p_3, p_4, p_5 are obtained.

Assume that the points within the double crosshatched region are in the high-speed storage. The computation scheme progresses as follows: We simultaneously compute $p_1, \cdots, p_5$ and place them in temporary storage. Next we want to compute the points p on the line $p \cdot x_2 = 61$, $60 \le p \cdot x_1 \le 74$. To this end we update the contents of the high-speed storage by storing in bulk storage the points on the line $p \cdot x_2 = 55$, which were previously in the high-speed storage, but are not needed as operands for computing the next five points, and fetching from bulk storage the points p satisfying $p \cdot x_2 = 60$, $55 \le p \cdot x_1 < 60$.

Successive computations are performed along $p \cdot x_2 = 62, 63$, and so on, until the whole "tube" of points $p \in F_2$ satisfying $60 \le p \cdot x_1 \le 74$ are computed. The scheme then progresses to another higher "tube"; namely, $75 \le p \cdot x_1 \le 89$. It is easily seen that using this scheme every point of F_2 will eventually be computed, and except for points too near the boundary exactly five points can be computed at a time.

This type of scheme is readily generalized to any function $a(p)$ explicitly defined over F_n. From (iii) of Theorem 2, we can always find n linearly independent vectors, with rational coefficients $x_1, x_2, \cdots, x_n$ such that $w_j \cdot x_k \ge 1$ for all $1 \le j \le s$, $1 \le k \le n$. We need $n - 1$ of these vectors to define the tube, and we assume that they are $x_1, x_2, \cdots, x_{n-1}$. The tube then is the set of points $p \in F_n$ such that $a_i \le p \cdot x_i \le b_i$, $i = 1, 2, \cdots, n - 1$. The set of points which will be computed simultaneously lie in the cross section defined by the intersection of the tube and $a_n \le$

$p \cdot x_n \leq b_n$. (In the example we used $b_2 = a_2$ and $b_1 = a_1 + 14$.) For a fixed selection of the separating hyperplanes, the number of points computed simultaneously is proportional (as a function of N) to $\pi = \prod_{i=1}^{n-1} (b_i - a_i)$, and the size of the high-speed storage is also proportional to π; in other words, the number of points computed simultaneously is directly proportional to the size of the high-speed storage.

The number of points which must be fetched from bulk storage to be used as operands when computing a cross section is proportional to

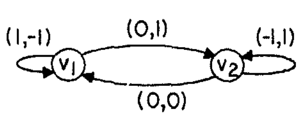
$$\pi' = \pi \sum_{j=1}^{n-1} \frac{1}{b_j - a_j}.$$

This follows from the fact that previously computed points must be fetched around the tube in all directions except the direction of propagation. For a given amount of parallelism, the amount of fetching from bulk storage can be reduced by making $b_i - a_i$ approximately equal for all i. Then the ratio of the number of points computed to the number of points fetched from bulk storage is proportional to $\sqrt[n-1]{N}$, where N is the size of the high-speed storage. Thus, as N increases, the average number of points fetched from bulk storage per point computed approaches zero as $1/\sqrt[n-1]{N}$.

5. *Systems of Recurrence Equations*

This section deals with systems of uniform recurrence equations over the domain F_n. Tests are derived to determine whether a system is explicitly defined over F_n, and the degree of parallelism in schedules for explicitly defined systems is investigated. A necessary and sufficient condition for a system to be explicitly defined has already been given in Theorem 1. For a single equation this condition reduces to the existence of a separating hyperplane (Theorem 2), which can be tested by linear programming. For systems, the testing process is more complicated. It involves an iterative decomposition of the dependence graph, requiring the solution of linear programs at each step. The investigation of parallelism is also more complicated for systems than for single equations. Whereas the free schedule for an explicitly defined single equation with $n > 1$ necessarily has unbounded parallelism, there exist explicitly defined systems with $n > 1$ for which the free schedule has bounded parallelism. This is illustrated in Example 6. Although we do not have general tests to distinguish whether a system has bounded or unbounded parallelism, two sufficient conditions for unbounded parallelism are given, and a conjecture about parallelism is made which involves the iterative decomposition of the dependence graph. Also, the problem of modifying a free schedule having bounded parallelism to reduce the maximum degree of parallelism while preserving a specified computation rate is considered.

$$a_1(p) = f_1(a_1(p - (1, -1)), a_2(p - (0, 1)))$$



$$a_2(p) = f_2(a_2(p - (-1, 1)), a_1(p - (0, 0)))$$

FIG. 5

Example 6. Since the dependence graph shown in Figure 5 does not have a nonpositive cycle, it follows from Theorem 1 that $a_1(p)$ and $a_2(p)$ are explicitly defined over F_2. For this example, only one point of $\{1, 2\} \times F_2$ can be computed at any time. This can be seen as follows:

$(1, (1, y)) \xrightarrow{1} (2, (1, y - 1))$, because of the edge labeled $(0, 1)$;

similarly,

$(1, (x, y)) \xrightarrow{1} (1, (x - 1, y + 1))$;
$(2, (x, 1)) \xrightarrow{1} (1, (x, 1))$;
$(2, (x, y)) \xrightarrow{1} (2, (x + 1, y - 1))$.

Thus, on a given line $x + y = k$, any schedule must compute the quantities $a_1(p)$ successively starting with $p = (1, k - 1)$ and ending with $p = (k - 1, 1)$; then the quantities $a_2(p)$ can be computed successively starting with $p = (k - 1, 1)$ and ending with $p = (1, k - 1)$. Only after $a_2(p)$ for $p = (1, k - 1)$ has been computed can the computation be started on the next line $x + y = k + 1$. Thus, in the free schedule, $\phi_T(1, \tau) + \phi_T(2, \tau) = 1$ for all τ; and every schedule S satisfies $\phi_S(1, \tau) + \phi_S(2, \tau) \leq 1$ for all τ.

Locating Nonpositive Cycles. A directed graph is called *strongly connected* if any two distinct vertices lie in a common cycle. A *strong component* of a directed graph is a strongly connected subgraph not properly contained in any other strongly connected subgraph. In general, a directed graph may contain several strong components, and each vertex lies in exactly one strong component.

Once it is known which strong components of a dependence graph contain nonpositive cycles, Theorem 1 can be applied to determine which functions $a_k(p)$ are explicitly defined. Thus, our procedure for locating nonpositive cycles treats each strong component of a dependence graph separately. In developing the procedure, the hypothesis of strong connectedness is explicitly stated when it is needed, since most of the results do not require it.

Let $C(G)$ denote the set of all integer n-vectors of the form $w(\pi)$, where π is a cycle of G; then our problem is to determine the nonpositive vectors, if any, in $C(G)$, as well as the corresponding nonpositive cycles. Let $L(G)$ denote the set of all semipositive integer combinations of vectors in $C(G)$; in fact, $L(G)$ is generated by the weights of the finite set of simple cycles of G. When G has a single vertex, $L(G) = C(G)$; otherwise, $L(G) \supseteq C(G)$, and the inclusion is usually proper. Thus, in Figure 5, $L(G)$ contains the vector $(1, -1) + (-1, 1) = (0, 0)$, but $C(G)$ does not. It is relatively easy to determine whether $L(G)$ contains a nonpositive vector; the difficulties in locating nonpositive cycles arise because $L(G)$ may be unequal to $C(G)$.

Lemma 4. *If $L(G)$ does not contain a nonpositive vector, then G does not contain a nonpositive cycle.*

Proof. $L(G) \supseteq C(G)$.

Lemma 5. *If G is strongly connected, then G contains a strictly negative cycle if and only if $L(G)$ contains a strictly negative vector.*

Proof. Let $V \in L(G)$ be a strictly negative vector. Then either $V = w(\gamma)$, where γ is a cycle of G, or

$$V = \sum_{i=1}^{r} \alpha_i w(\gamma_i),$$

where $r > 1$ and, for each i, α_i is a positive integer and γ_i is a cycle of G. If

$V = w(\gamma)$, the result follows. Otherwise, because G is strongly connected, there is a cycle γ_0 which includes all the vertices of G, and thus every vector of the form

$$w(\gamma_0) + \sum_{i=1}^{r} \beta_i w(\gamma_i),$$

where the β_i are positive integers, is the weight of a cycle of G. Choosing $\beta_i = K\alpha_i$ for sufficiently large K gives a strictly negative cycle. Thus G contains a strictly negative cycle if $L(G)$ contains a strictly negative vector. The converse statement follows from the fact that $L(G) \supseteq C(G)$.

Motivated by Lemmas 4 and 5, we present tests for the existence of negative vectors or nonpositive vectors in $L(G)$. These tests are based on the following theorem.

THEOREM 6. *A vector V is an element of $L(G)$ if and only if*

$$V = \sum_{i=1}^{s} u_i w_i,$$

where the coefficients u_i satisfy:

(i) *u_i is a nonnegative integer;*

(ii) $\sum_{i=1}^{s} u_i \geq 1$;

(iii) $\sum_{i \in I_k(G)} u_i - \sum_{i \in O_k(G)} u_i = 0, \quad k = 1, 2, \cdots, m,$

where $I_k(G) = \{i \mid v_k$ is the vertex to which edge e_i is directed$\}$ and $O_k(G) = \{i \mid v_k$ is the vertex from which edge e_i is directed$\}$.

PROOF. If γ_1 is a cycle of G, then

$$w(\gamma_1) = \sum_{i=1}^{s} u_i^{(1)} w_i,$$

where $u_i^{(1)}$ denotes the number of occurrences of e_i in γ_1. Clearly, $(u_1^{(1)}, u_2^{(1)}, \cdots, u_s^{(1)})$ satisfies (i)–(iii). If V is in $L(G)$, then

$$V = \sum_{j=1}^{r} \alpha_j w(\gamma_j) = \sum_{j=1}^{r} \sum_{i=1}^{s} \alpha_j u_i^{(j)} w_i,$$

and setting

$$u_i = \sum_{j=1}^{r} \alpha_j u_i^{(j)},$$

we see that $(u_1, u_2, \cdots, u_s)$ satisfies (i)–(iii). Thus the given conditions are necessary in order that V be in $L(G)$.

The proof of sufficiency is a direct consequence of a lemma of I. J. Good [4], which states that if the edges of a connected graph are oriented so that at each vertex the number inward is equal to the number outward, then the whole graph can be traversed as a closed circuit in which each edge occurs once and only once and with its assigned orientation.

From Theorem 6 we obtain the following corollary.

COROLLARY 4. *Let V be an integer n-vector and E a subset of $\{1, 2, \cdots, s\}$. Then the following statements are equivalent:*

(1) *V can be expressed as*

$$V = \sum_{i=1}^{s} u_i w_i,$$

where $(u_1, u_2, \cdots, u_s)$ *satisfies* (i)–(iii) *of Theorem* 6, *and* $u_i > 0$ *if and only if* $i \in E$;

(2) *V can be expressed as*

$$V = \sum \alpha_j w(\gamma_j),$$

where the α_j *are positive integers, the* γ_j *are cycles of* G, *and* e_i *is an edge of some cycle* γ_j *if and only if* $i \in E$.

Adding to the inequalities of Theorem 6 the requirements that the vector be, respectively, nonpositive or negative, we obtain the following two corollaries.

COROLLARY 5. *$L(G)$ contains a nonpositive vector if and only if the following system of linear inequalities has a feasible solution:*

$$\text{III} \quad \begin{cases} u_i \geq 0, & i = 1, 2, \cdots, s, \\ \sum_{i=1}^{s} u_i > 0, & \\ \sum_{i \in I_k(G)} u_i - \sum_{i \in O_k(G)} u_i = 0, & k = 1, 2, \cdots, m, \\ \sum_{i=1}^{s} (u_i w_i)_j \leq 0, & j = 1, 2, \cdots, n. \end{cases}$$

COROLLARY 6. *$L(G)$ contains a strictly negative vector if and only if the following system of linear inequalities has a feasible solution:*

$$\text{IV} \quad \begin{cases} u_i \geq 0, & i = 1, 2, \cdots, s, \\ \sum_{i=1}^{s} u_i > 0, & \\ \sum_{i \in I_k(G)} u_i - \sum_{i \in O_k(G)} u_i = 0, & k = 1, 2, \cdots, m, \\ \sum_{i=1}^{s} (u_i w_i)_j \leq -1, & j = 1, 2, \cdots, n. \end{cases}$$

From these results we conclude that if system III has no feasible solution, then G does not contain a nonpositive cycle, and, if system IV has a feasible solution, then, provided G is strongly connected, G contains a strictly negative cycle. The feasibility of systems III and IV can be tested by standard methods. Thus, the only case requiring further analysis in determining whether a strong component G of a dependence graph contains a nonpositive cycle occurs when system III has a feasible solution, but system IV does not.

To investigate this case further, we define G', a subgraph of G. The edge e_i of G is also an edge of G' if and only if system III has a feasible solution in which $u_i > 0$.[2] Every edge of G' has the same weight as it had in G, and the vertices of G' are those vertices of G which are incident with edges of G'.

THEOREM 7. *The subgraph G' has the following properties:*

(1) *e_i is an edge of G' if and only if e_i is contained in some semipositive combination of cycles of G whose total weight is nonpositive;*

[2] The set of edges of G' can be obtained by applying the Simplex method of linear programming to system III with an appropriately chosen sequence of objective functions.

(2) *G contains a nonpositive cycle if and only if G' does;*
(3) *G' is a union of disjoint strongly connected graphs;*
(4) *if G' is strongly connected, then G contains a nonpositive cycle.*

PROOF. Statement (1) follows directly from Corollary 4 and the definition of G', and clearly (2) is immediate from (1). From (1), every edge of G' is in some cycle of G'; thus, (3) holds. To prove (4), we assume G' is strongly connected and display a nonpositive cycle of G' (hence, also, of G, since every cycle of G' is also a cycle of G). Since G' is strongly connected, there is a cycle C passing through all the vertices of G'. Let the successive edges of C be $d_1, d_2, \cdots, d_r$. Then, since (1) has been established, each edge d_i is contained in a semipositive combination of cycles of G' with nonpositive total weight. The combination corresponding to d_i may be expressed as

$$w(\gamma_i) + \sum \alpha_{ij} w(\delta_{ij}) \leq 0,$$

where γ_i is a cycle containing d_i, the α_{ij} are nonnegative integers, and the δ_{ij} are cycles of G'. Moreover, since γ_i contains d_i, γ_i may be decomposed into two parts: d_i and π_i, a path containing the remaining edges of γ_i and directed from the vertex to which d_i is directed to the vertex from which d_i is directed. We can now construct a cycle with weight

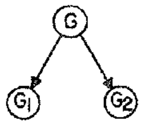
$$\sum_i [w(\gamma_i) + \sum_j \alpha_{ij} w(\delta_{ij})] \leq 0$$

as follows. The cycle traverses the edges $d_1, \cdots, d_r$ in turn, and then returns along the paths $\pi_r, \pi_{r-1}, \cdots, \pi_1$. Since each vertex is visited in this process, it is possible to insert α_{ij} traversals of each cycle δ_{ij} at some convenient point. This produces the required nonpositive cycle.

Theorem 7 enables us to construct an iterative procedure for determining the nonpositive cycles of G. Recall that, in the case under consideration, G has no strictly negative cycles, and $L(G)$ contains a nonpositive vector. The first step is to construct G'. If G' is strongly connected, then, by Theorem 7, G contains a nonpositive cycle. Furthermore, the construction used to prove Theorem 7 constructs a nonpositive cycle passing through all the vertices of G'. If G' is not strongly connected, then it is a union of disjoint strongly connected components $G_1, G_2, \cdots, G_t$, and each nonpositive cycle of G must be a nonpositive cycle of one of these components. The components are then "decomposed" to obtain subgraphs $G_1', G_2', \cdots, G_{t'}'$ in the same manner as G' was derived from G, and the iterative process continues until either a nonpositive cycle is located or no further decomposition is possible.

The entire decomposition can be represented by a directed tree whose vertices correspond to the original dependence graph G and the strong components arising in the decomposition. The tree is rooted at the vertex corresponding to G, and, for each subgraph H which occurs, there is an edge directed from the vertex corresponding to H to the vertices corresponding to $H_1, H_2, \cdots, H_t$, the components of H'. The number of vertices in the longest path in this tree is called the *depth* of the decomposition of G.

Example 7. Let G be the graph of Figure 5. For this graph, system IV has no feasible solution, but system III does. The subgraph G' takes the form shown in Figure 6. For each of G_1 and G_2, system III has no feasible solution, so G_1' and G_2' are null. The decomposition process terminates at this point, and G has been found to have no nonpositive cycles. Thus, by Theorem 1, $a_1(p)$ and $a_2(p)$ are explicitly defined over F_2. In this case, the decomposition tree is simply



and it has depth 2.

PARALLELISM. As was shown in Example 6, it is possible that the free schedule for a fully defined system of recurrence equations over $F_n(n > 1)$ may not have unbounded parallelism. This is an important qualitative difference between single equations and systems of equations, and it suggests the general problem of characterizing dependence graphs with bounded parallelism. This problem has not been solved completely, but we present sufficient conditions for unbounded parallelism. Although our sufficient conditions apply to general dependence graphs, their main application is to individual strong components. The relationship between the parallelism in the free schedule for a dependence graph and the parallelism in the free schedules of its strong components is discussed after the sufficient conditions are given.

The sufficient conditions depend on the depth of the decomposition tree for G. We prove that, if the depth of the decomposition of G is less than n, and if certain further algebraic conditions are met, then the free schedule for G has unbounded parallelism. We turn next to the formulation of these conditions.

LEMMA 6. *Let G be a dependence graph. Then there exist real numbers $\rho_1, \rho_2, \cdots, \rho_m$ and a nonnegative n-vector x such that, for any edge e_i, directed from v_k to v_l,*

$$w_i \cdot x \geq \rho_l - \rho_k,$$

and, if $e_i \notin G'$, then

$$w_i \cdot x \geq 1 + \rho_l - \rho_k.$$

PROOF. By definition of G', system III does not have a feasible solution such that

$$\sum_{\{i \mid e_i \notin G'\}} u_i > 0.$$

The desired result follows by applying the Minkowski-Farkas lemma to the following system of inequalities:

$$\begin{cases} \text{(i)} \quad u_i \geq 0, & i = 1, 2, \cdots, s, \\ \text{(ii)} \quad \sum_{\{i \mid e_i \notin G'\}} u_i = 1, & \\ \text{(iii)} \quad \sum_{i \in I_k(G)} u_i - \sum_{i \in O_k(G)} u_i = 0, & k = 1, 2, \cdots, m, \\ \text{(iv)} \quad \sum_{i=1}^{s} (u_i w_i)_j \leq 0, & j = 1, 2, \cdots, n. \end{cases}$$

The variable ρ_k corresponds to the kth equation in (iii), the variable x_j to the jth constraint of (iv), and d to the right-hand side of (ii).

FIG. 6

Any vector x which, for some choice of $\rho_1, \rho_2, \cdots, \rho_m$, satisfies Lemma 6, is called a *weakly separating hyperplane* for G.

COROLLARY 7. *Let G, $\rho_1, \rho_2, \cdots, \rho_m$ and x be as in Lemma* 6. *If there is a path in G of weight w from v_k to v_l containing N occurrences of edges not in G', then $w \cdot x \geq N - (\rho_k - \rho_l)$. In particular, $w \cdot x \geq N - K$, where $K = max_{k,l}(\rho_k - \rho_l)$.*

THEOREM 8. *Let G be a dependence graph such that every function $a_k(p)$ is explicitly defined over F_n. If the decomposition of G has depth $r < n$, and every strong component occurring in the decomposition has a weakly separating hyperplane which is strictly positive, then, for every k, $sup_\tau\, \phi_T(k, \tau) = \infty$.*

The proof of Theorem 8 requires the following lemma.

LEMMA 7. *Let G be a dependence graph which satisfies the hypotheses of Theorem* 8. *If x is a weakly separating hyperplane which is strictly positive, then there exists a positive constant α such that, for all $(k, p) \in \{1, 2, \cdots, m\} \times F_n$,*

$$T(k, p) \leq \alpha(p \cdot x)^r.$$

PROOF. The proof is by induction on r. When $r = 1$, G' is empty. There exists (l, q) such that $q \in F_n$ and $(k, p) \xrightarrow{t} (l, q)$, where $t = T(k, p) - 1$. By Corollary 7,

$$(p - q) \cdot x \geq t - K;$$

therefore,

$$p \cdot x \geq (p - q) \cdot x \geq t - K = T(k, p) - (K + 1).$$

Thus,

$$T(k, p) \leq p \cdot x + (K + 1),$$

and the result follows.

Now assume that the result has been proved for all $c < r$. Since the decomposition of G has depth r, the decomposition of each of the components $G_1, G_2, \cdots, G_t$ of G' has depth less than r. Also, the hypotheses of the lemma apply to these components and their decompositions. Thus, if y_h is a strictly positive weakly separating hyperplane for G_h, then, denoting by T_h the free schedule for G_h, $T_h(k, p) \leq \alpha_h(p \cdot y_h)^{c_h}$, where $c_h (c_h < r)$ is the depth of the decomposition of G_h. Since x and y_h are both positive vectors, there is a constant $\beta_h = \max_i ((y_h)_i / x_i)$ such that $p \cdot y_h \leq \beta_h(p \cdot x)$ for all $p \in F_n$. Therefore, $T_h(k, p) \leq \alpha_h \beta_h^{c_h}(p \cdot x)^{c_h} = \alpha_h'(p \cdot x)^{c_h}$. By the definition of $T(k, p)$ there exists an (l, q) such that $q \in F_n$ and $(k, p) \xrightarrow{t} (l, q)$, where $t = T(k, p) - 1$. Let the corresponding path of length t contain N occurrences of edges not in G' joining together at most $N + 1$ subpaths $\pi_1, \pi_2, \cdots, \pi_u$, each entirely within a component of G'. Clearly

$$t = N + \sum_{i=1}^{u} l(\pi_i).$$

We bound $T(k, p)$ by obtaining bounds on N and each $l(\pi_i)$. By Corollary 7, $(p - q)\cdot x \geq N - K$, so $N \leq (p - q)\cdot x + K \leq (p\cdot x) + K$. Similarly, if (l_i, q_i) is the starting point of the subsequence corresponding to path π_i, then

$$0 \leq (p - q_i)\cdot x + K,$$

so

$$q_i\cdot x \leq p\cdot x + K.$$

Hence, if π_i lies in G_h, then

$$l(\pi_i) \leq T_h(l_i, q_i) \leq \alpha_h'(q_i\cdot x)^{c_h} \leq \alpha_h'(p\cdot x + K)^{c_h}.$$

Hence, there exists a suitably large constant γ such that

$$l(\pi_i) \leq \gamma(p\cdot x)^{r-1}, \qquad i = 1, 2, \cdots, u.$$

Substituting, we have

$$\begin{aligned} T(k, p) = t + 1 = 1 + N + \sum_{i=1}^{u} l(\pi_i) &\leq 1 + (p\cdot x) + K + u\gamma(p\cdot x)^{r-1} \\ &\leq 1 + (p\cdot x) + K + (p\cdot x + K + 1)\gamma(p\cdot x)^{r-1} \\ &\leq \alpha(p\cdot x)^r \end{aligned}$$

for suitably large α.

PROOF OF THEOREM 8. Choose k, $1 \leq k \leq m$. By Lemma 7,

$$\begin{aligned} \sum_{\tau=1}^{\theta} \phi_T(k, \tau) = \|\{p \mid \mathrm{T}(k, p) \leq \theta\}\| &\geq \|\{p \mid \alpha(p\cdot x)^r \leq \theta\}\| \\ &= \|\{p \mid p\cdot x \leq \sqrt[r]{(\theta/\alpha)}\}\| = O(\theta^{n/r}). \end{aligned}$$

Since $n > r$, so that $n/r > 1$, there is no constant which uniformly bounds $\phi_T(k, \tau)$, so that $\sup_\tau \phi_T(k, \tau) = \infty$ for every k.

In the special case where G' is empty ($r = 1$) there exists a "strictly separating hyperplane" x just as in the case of a single equation. That is, there exist real numbers $\rho_1, \rho_2, \cdots, \rho_m$ and a nonnegative vector x such that, for any edge e_i, directed from v_k to v_l

$$w_i\cdot x \geq 1 + \rho_l - \rho_k .$$

It can be shown that $\rho_1, \rho_2, \cdots, \rho_m$ can actually be chosen as integers (with an appropriate choice of x) so that such a system can be transformed into a single explicitly defined uniform recurrence equation in the vector-valued variable $b(p)$, where

$$b_k(p) = a_k(p - p_k) \quad \text{and} \quad p_k\cdot x = \rho_k , \qquad k = 1, 2, \cdots, m.$$

Of particular interest is the fact that this transformation enables us to apply the storage scheme of Section 4 to such systems.

Example 8. Consider the system

$$\begin{aligned} a_1(p) &= f_1(a_1(p - (0, 1)), a_2(p - (-1, -1))), \\ a_2(p) &= f_2(a_1(p - (3, 2)), a_2(p - (1, 0))). \end{aligned}$$

The dependence graph is shown in Figure 7. Clearly $L(G)$ contains no nonpositive vector. Setting $\rho_1 = 4$, $\rho_2 = 0$, and $x = (1, 1)$ we find that:

$$(0, 1)\cdot x \geq 1, \qquad (-1, -1)\cdot x \geq 1 + 0 - 4,$$
$$(1, 0)\cdot x \geq 1, \qquad (3, 2)\cdot x \geq 1 + 4 - 0.$$

Setting $p_1 = (2, 2)$ and $p_2 = (0, 0)$, the conditions $p_k \cdot x = \rho_k$ are satisfied. Making the transformation

$$b_k(p) = a_k(p - p_k),$$

we obtain the system

$$b_1(p) = f_1(b_1(p - (0, 1)), b_2(p - (1, 1))),$$
$$b_2(p) = f_2(b_1(p - (1, 0)), b_2(p - (1, 0))).$$

Writing

$$b(p) = \begin{pmatrix} b_1(p) \\ b_2(p) \end{pmatrix},$$

we have

$$b(p) = f(b(p - (0, 1)), b(p - (1, 1)), b(p - (1, 0))),$$

and a single explicitly defined equation has been obtained.

The results we have obtained do not completely characterize the strongly connected dependence graphs whose free schedules have bounded parallelism. Of course, every strongly connected dependence graph with at least one edge has bounded parallelism in the case $n = 1$. Furthermore, Example 6 shows that if $m > 1$, graphs with bounded parallelism exist even when $n > 1$. Also, Theorem 8 gives sufficient conditions for a strongly connected graph to have unbounded parallelism.

From Theorem 8 one might suspect that, when the decomposition of G has depth n (it can be shown that the decomposition never has depth greater than n), the free schedule has bounded parallelism. The following slight modification of Example 6 shows that this is false.

Example 9. In Figure 8, the decomposition is exactly as in Example 6 and has depth 2, but $T(1, p) = 1$ for all p of the form $(1, y)$, so that $\phi_T(1, 1) = \infty$. We believe, however, that this type of unbounded parallelism is limited to "corridors" near the boundary of F_n. More precisely, we make the following conjecture.

Conjecture. Let G be a strongly connected dependence graph such that every function $a_k(p)$ is explicitly defined over F_n. If the decomposition of G has depth $r < n$, then, for every k, $\sup_\tau \phi_T(k, \tau) = \infty$. If the decomposition of G has depth n, then there exists a point $\pi \in F_n$ such that, for each k, $\max_\tau \|\{p \mid p \geq \pi \text{ and } T(k, p) = \tau\}\| < \infty$.

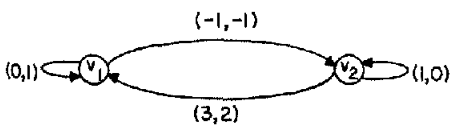


FIG. 7

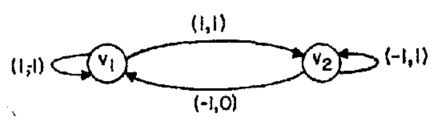


FIG. 8

In general, a dependence graph G will have several strong components. In some cases, properties of the free schedule for G can be determined from properties of the free schedules of its strong components. If there is a path from a vertex v_k in G to a strong component whose free schedule has bounded parallelism, then $\phi_T(k, \tau)$ is bounded. If, for every strong component reachable from v_k, an inequality of the form

$$T(k, p) \leq \alpha(p \cdot x)^r, \qquad r \leq n - 1 \tag{$*$}$$

holds, then $\phi_T(k, \tau)$ is unbounded. If G_1 is a strong component of G from which no other strong component can be reached, then the free schedules for G and G_1 are identical for vertices in G_1. If every strong component that can be reached from v_k has a free schedule with unbounded parallelism, but some of these strong components do not satisfy an inequality of the form ($*$), then our results do not establish whether $\phi_T(k, \tau)$ is bounded.

When the free schedule for a dependence graph has bounded parallelism, one can define its *maximum parallelism*,

$$M(T) = \max_{\tau} \sum_{k} \phi_T(k, \tau),$$

and its *average parallelism*,

$$A(T) = \limsup_{\tau \to \infty} \frac{1}{t} \sum_{\tau=1}^{t} \sum_{k} \phi_T(k, \tau).$$

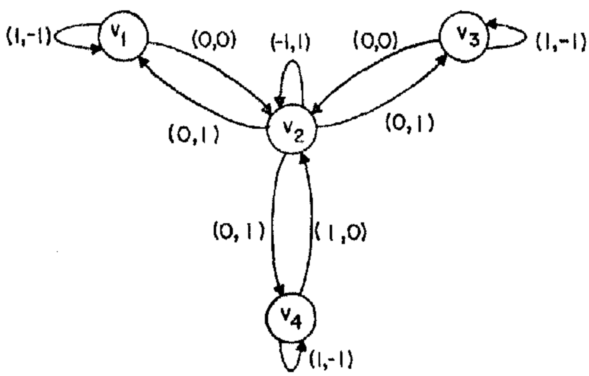


FIG. 9

TABLE 1

τ	T			S	
1	$a_1(1, 1)$	$a_3(1, 1)$	$a_4(1, 1)$	$a_1(1, 1)$	$a_3(1, 1)$
2		$a_2(1, 1)$		$a_2(1, 1)$	$a_4(1, 1)$
3	$a_1(1, 2)$	$a_3(1, 2)$	$a_4(1, 2)$	$a_1(1, 2)$	$a_3(1, 2)$
4	$a_1(2, 1)$	$a_3(2, 1)$	$a_4(2, 1)$	$a_1(2, 1)$	$a_3(2, 1)$
5		$a_2(2, 1)$		$a_2(2, 1)$	$a_4(1, 2)$
6		$a_2(1, 2)$		$a_2(1, 2)$	$a_4(2, 1)$
7	$a_1(1, 3)$	$a_3(1, 3)$	$a_4(1, 3)$	$a_1(1, 3)$	$a_3(1, 3)$
8	$a_1(2, 2)$	$a_3(2, 2)$	$a_4(2, 2)$	$a_1(2, 2)$	$a_3(2, 2)$
9	$a_1(3, 1)$	$a_3(3, 1)$	$a_4(3, 1)$	$a_1(3, 1)$	$a_3(3, 1)$
10		$a_2(3, 1)$		$a_2(3, 1)$	$a_4(1, 3)$

The implementation of the free schedule requires a computer capable of performing $M(T)$ simultaneous function evaluations. When $A(T) < M(T)$, however, the question arises whether there exists a schedule S which proceeds at the same average rate $(A(S) = A(T))$, but which can be implemented on a computer with fewer arithmetic units $(M(S) < M(T))$. The following examples show that this is sometimes possible.

Example 10. See Figure 9. The free schedule T and a "smoothed" schedule S are illustrated in this example. $M(T) = 3 > M(S) = 2$, but $A(T) = A(S) = 2$. The points calculated by S and T for the first 10 time units are shown in Table 1.

In this example, the schedule S is obtained by delaying the computation of points $a_4(p)$ by longer and longer amounts. If the example is modified so that the edge from v_4 to v_2 is labeled $(0, 0)$, then the free schedule is unchanged, but no schedule S exists such that $M(S) < M(T)$ but $A(S) = A(T)$. In the next example, smoothing is achieved without delaying any computation more than one unit of time.

Example 11. See Figure 10 and Table 2.

6. *Implicit Schedules*

Although many numerical methods of solving partial differential equations, such as point overrelaxation for Poisson's equation, lead to uniform recurrence equations

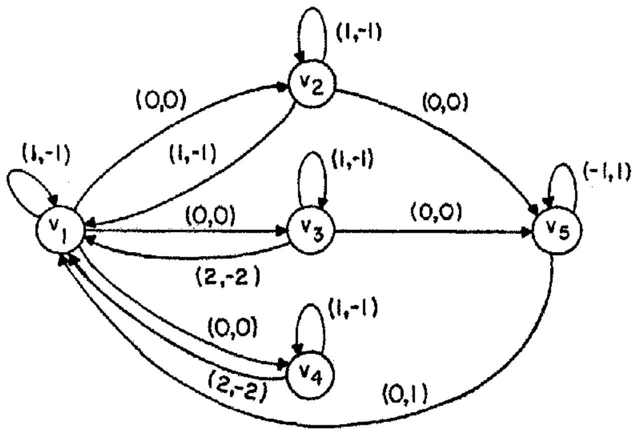


FIG. 10

TABLE 2

τ	T	S
1	$a_1(1, 1)$	$a_1(1, 1)$
2	$a_2(1, 1)$ $a_3(1, 1)$ $a_4(1, 1)$	$a_2(1, 1)$ $a_3(1, 1)$
3	$a_5(1, 1)$	$a_4(1, 1)$ $a_5(1, 1)$
4	$a_1(1, 2)$	$a_1(1, 2)$
5	$a_2(1, 2)$ $a_3(1, 2)$ $a_4(1, 2)$	$a_2(1, 2)$ $a_3(1, 2)$
6	$a_1(2, 1)$	$a_4(1, 2)$ $a_1(2, 1)$
7	$a_2(2, 1)$ $a_3(2, 1)$ $a_4(2, 1)$	$a_2(2, 1)$ $a_3(2, 1)$
8	$a_5(2, 1)$	$a_4(2, 1)$ $a_5(2, 1)$
9	$a_5(1, 2)$	$a_5(1, 2)$
10	$a_1(1, 3)$	$a_1(1, 3)$
11	$a_2(1, 3)$ $a_3(1, 3)$ $a_4(1, 3)$	$a_2(1, 3)$ $a_3(1, 3)$

which are explicitly defined in the sense of the previous sections, certain important methods, such as line relaxation, lead to equations for which no schedule exists. These so-called implicit methods require the repeated solution of subsystems of simultaneous equations. In this section we model the structure common to these methods by introducing the concept of an implicit schedule. For the sake of contrast, the schedules discussed up to now are referred to as explicit schedules in the remainder of this section.

Let G be a dependence graph for a system of m uniform recurrence equations defined over the region R. An *implicit schedule* S is a function from $\{1, 2, \cdots, m\} \times R$ into the positive integers satisfying the following condition: For each positive integer τ, $\{(k, p) \mid S(k, p) = \tau\}$ can be partitioned into finite subsets $\Sigma_1(\tau)$, $\Sigma_2(\tau)$, $\cdots$ such that, if $(k, p) \in \Sigma_i(\tau)$ and $(k, p) \xrightarrow{1} (l, q)$, then either

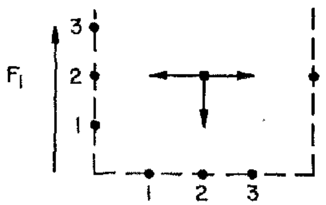
(i) $S(l, q) < \tau$, or

(ii) $(l, q) \in \Sigma_i(\tau)$.

Note that every explicit schedule is also an implicit schedule in which each set $\Sigma_i(\tau)$ may be taken to consist of a single element. Also, the following example shows that an implicit schedule can exist even when no explicit schedule exists.

Example 12. Let $R = \{1, 2, 3\} \times F_1$, and consider a uniform recurrence equation of the form

$$a(x, y) = f(a(x - 1, y), a(x + 1, y), a(x, y - 1)).$$



Then no explicit schedule exists, but the function $S(p) = y$, where $p = (x, y)$, is an implicit schedule. This implicit schedule corresponds to a computation in which a system of three simultaneous equations is solved at each time y.

This section is concerned with formulating some necessary and sufficient conditions for the existence of an implicit schedule. We first derive necessary and sufficient conditions applicable to arbitrary regions, and then specialize them to the case that the region is F_n or $Q_t \times F_{n-t}$. Since the proofs of the theorems of the section are analogous to the proofs of the corresponding theorems for explicit schedules, we omit all proofs.

Consider the following equivalence relation "$\sim$" over $\{1, 2, \cdots, m\} \times R$:

(i) $(k, p) \sim (k, p)$;

(ii) $(k, p) \sim (l, q)$ if and only if $(k, p) \to (l, q)$ and $(l, q) \to (k, p)$, for $(k, p) \neq (l, q)$.

It is clear that two distinct elements (k, p) and (l, q) are equivalent if and only if they are mutually dependent. The concept of dependence can be extended to dependence between equivalence classes. Let $[(k, p)]$ denote the equivalence class which includes (k, p). We say that $[(k, p)] \to [(l, q)]$ if and only if $(k, p) \to (l, q)$. It is clear that this definition is independent of the choice of elements which represent the equivalence classes.

THEOREM 9. *Let $\{f_i\}$ be a system of uniform recurrence equations over R. The following four statements are equivalent:*

(i) *there exists an implicit schedule;*

(ii) *every equivalence class of "$\sim$" is finite and depends on only a finite number of equivalence classes;*

(iii) *for each $(k, p) \in \{1, 2, \cdots, m\} \times R$ the set $\{(l, q) | (k, p) \rightarrow (l, q)\}$ is finite;*

(iv) *there does not exist an infinite sequence of distinct elements $\{(k_i, p_i)\}$ such that $(k_j, p_j) \xrightarrow{1} (k_{j+1}, p_{j+1})$.*

In the case that R is F_n we have the following additional necessary and sufficient condition for an implicit schedule to exist.

THEOREM 10. *Let G be the dependence graph of a system of uniform recurrence equations defined over F_n. There exists an implicit schedule if and only if G does not contain a cycle whose weight is a seminegative vector (i.e., a nonzero vector none of whose coordinates are positive).*

COROLLARY 8. *Let G be the dependence graph of a system of uniform recurrence equations defined over $Q_t \times F_{n-t}$, where Q_t is a finite subset of F_t. Assume that no explicit schedule exists. An implicit schedule exists if and only if G contains no cycle whose weight is a seminegative vector whose first t coordinates are zero.*

Specializing further to the case of a single equation ($m = 1$), we obtain:

THEOREM 11. *Consider the equation $a(p) = f(a(p - w_1), a(p - w_2), \cdots, a(p - w_s))$, defined over F_n. An implicit schedule exists for this equation if and only if there exists a positive n-vector x such that $x \cdot w_i \geq 0$, $i = 1, 2, \cdots, s$.*

COROLLARY 9. *Let $a(p) = f(a(p - w_1), a(p - w_2), \cdots, a(p - w_s))$ be an equation over $Q_t \times F_{n-t}$. Assume the equation has no explicit schedule. Then the equation has an implicit schedule if and only if there exists an n-vector x which is positive in its last $n - t$ coordinates such that $x \cdot w_j \geq 0$, $j = 1, 2, \cdots, s$.*

COROLLARY 10. *If $a(p)$ has an implicit schedule but not an explicit schedule over $Q_t \times F_{n-t}$ then it has an implicit schedule over $Q_t' \times F_{n-t}$ for all finite sets $Q_t' \subseteq F_t$.*

Every n-vector x satisfying the conditions of Theorem 11 is the normal vector to a unique $(n - 1)$-dimensional hyperplane through the origin. Let the linear subspace of dimension k which is the intersection of all such hyperplanes be denoted by V_k. Then an explicit schedule exists if and only if $k = 0$. Let a congruence relation "$\equiv$" between points of F_n be defined as follows: $p \equiv q$ if and only if $p - q \in V_k$. Then the equivalence relation "$\sim$" is a refinement of "$\equiv$". The image of F_n in the quotient space E_n/V_k is isomorphic to F_{n-k}, and a reduced uniform recurrence structure over F_{n-k} is determined by

$$h(p) \xrightarrow{1} h(q) \quad \text{if and only if} \quad h(p) \neq h(q) \quad \text{and} \quad p \xrightarrow{1} q,$$

where h is the projection homomorphism. In the reduced system an explicit schedule exists, and every such schedule defines an implicit schedule S' in the original system according to the relation $S'(p) = S(h(p))$. Also, the results of Section 4 apply to the reduced system. Thus, for example, if $k < n - 1$, so that the dimension of $E_n/V_k > 1$, the free schedule T for the reduced system has unbounded parallelism, and it follows that, in the corresponding implicit schedule T', there is no bound on the number of equivalence classes on which computation is done simultaneously.

ACKNOWLEDGMENT. The authors wish to express thanks to Dr. J. Cocke for point-

ing out the relevance of [7], to the referee for improving the proof of Theorem 1, and to Dr. D. E. Knuth for his careful reading and criticism of the manuscript.

REFERENCES

1. BERGE, C. *The Theory of Graphs and Its Applications.* Wiley, New York, 1962.
2. DANTZIG, G. B. *Linear Programming and Extensions.* Princeton U. Press, Princeton, N. J., 1963.
3. GOMORY, R. E. On the relation between integer and noninteger solutions to linear programs. *Proc. Nat. Acad. Sci. 53*, 2 (Feb. 1965), 260–265.
4. GOOD, I. J. Normal recurring decimals. *J. London Math. Soc. 21* (1946), 167–169.
5. JEENEL, J. Programs as a tool for research in systems organization. *IBM J. 2*, 2 (April 1958), 105–122.
6. KARP R. M., AND MILLER, R. E. Properties of a model for parallel computations: determinacy, termination, queueing. *SIAM J. Appl. Math. 14*, 6 (Nov. 1966), 1390–1411.
7. THOMAS, J. M. Orderly differential systems. *Duke Math. J. 7* (Dec. 1940), 249–290.

RECEIVED AUGUST, 1966; REVISED JANUARY, 1967