DISCUSSION PAPER NO. 254

FIXED POINT COMPUTING METHODS

by

ROMESH SAIGAL */

November 1976

*/
Department of Industrial Engineering
Northwestern University
Evanston, Illinois 60201

Fixed Point Computing Methods

by

R. Saigal

ABSTRACT

This paper describes the recent algorithms, based on complementary pivoting, designed to compute fixed points of certain upper semi-continuous point to set mappings in finite dimensional spaces, and has been written for the Encyclopedia of Computer Science and Technology, published by Marcel Dekker, Inc., New York.

Fixed Point Computing Methods*

by

R. Saigal

1. INTRODUCTION

Let R^n be the space of all n-dimensional vectors $x = (x_1,\ldots,x_n)$, (the n-dimensional Euclidean space) and let

$$f(x_1,\ldots,x_n) = \{f_1(x_1,\ldots,x_n),\ldots,f_n(x_1,\ldots,x_n)\}$$

be a function from R^n into R^n, i.e., the i^{th} component of this function is a real valued function $f_i(x_1,\ldots,x_n)$ of n-variables $x_1,\ldots,x_n$. Then, the problem we consider is to compute a vector x in R^n such that

$$f(x) = x \tag{1.1}$$

that is, x is a fixed point of f. This problem is closely related to solving a system of n equations in n variables, that is computing a vector x in R^n such that

$$f(x) = y$$

---

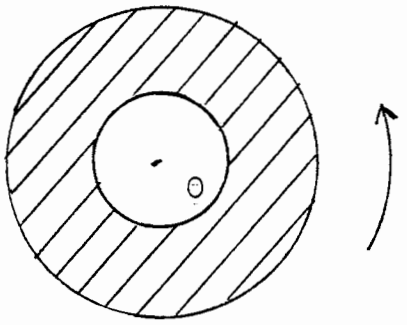
* This article has been written for the Encyclopedia of Computer Science and Technology, published by Marcel Dekker, Inc., New York.

where y is some given vector in R^n. The traditional methods for solving (1.1) generally require the differentiability of f, or some rather strong property like contractability. In this article, we will consider methods based on complementary pivoting, a concept introduced by Lemke and Howson [9], Lemke [11]; and refer the reader to the excellent book, Ortega and Rhheinboldt [13], for a complete study of the traditional methods. As we will subsequently see, the methods of complementary pivoting seem to require the minimal assumptions on f, and in many cases the conditions on f under which the existence of a solution has been established are sufficient for these methods to successfully compute a solution. We now introduce two such theorems.

Probably, the most celebrated fixed point theorem is that of Brouwer [2]. A subset S of R^n is called an n-dimensional simplex if it is a convex hull of n + 1 affinely independent points $v^1, v^2, \ldots, v^{n+1}$. The points v^i are called vertices of the simplex. (A set of n+1 points $v^1, \ldots, v^{n+1}$ in R^n are affinely independent if the $(n+1)\times(n+1)$ matrix $\begin{bmatrix} v^1 & \cdots & v^{n+1} \\ 1 & & 1 \end{bmatrix}$ has full rank. In two dimensions, three points are affinely independent if they are not colinear). A 2-dimensional simplex is a triangle and a 3-dimensional one a tetrahedron. We now state the theorem in its simplest form:

Brouwer's Theorem: Let S be an n-dimensional simplex, and let f be a continuous mapping from S into S. Then, there is an x in S such that f(x) = x; i.e. f has a fixed point x in S.

The above theorem holds under the relaxed assumption that S is a compact, convex subset of R^n. To see the need for the assumption of convexity in 2-dimensions, it is sufficient to note that the rotation of a ring (see Fig. 1.1) is a continuous mapping of the ring onto itself, which has no fixed points.



rotate the ring around the center O, and note that O is not in the ring.

Figure 1.1

The need for boundedness follows from the fact that the mapping $x + a$ $(a \neq 0)$ of R^n into R^n has no fixed points; and, for closedness is illustrated by the mapping $f(x_1,x_2) = \frac{1}{2}\{(x_1,x_2) + (1,0)\}$ of the open set $\{(x_1,x_2): |x_1| + |x_2| < 1\}$ onto itself whose only fixed point is (1,0).

The algorithm based on complementary pivoting, developed by Scarf [17] in 1967 was primarily motivated by a desire to compute a fixed point guaranteed by the Brouwer's theorem. He showed that for continuous f and any $\varepsilon > 0$, sufficiently small, the algorithm could be used to compute an approximate fixed point x, in the sense that $||f(x) - x|| \leq \varepsilon$.

Hansen [6] and Kuhn [8] related the procedure of Scarf to pivoting on a subdivision of the simplex, and thus considerably simplified the implementation and discussion. A major drawback of this algorithm is that a substantial amount of computational effort is involved before a reasonable approximation is reached. In addition, the computation has to be initiated at a point on the boundary of the simplex. In subsequent works, Eaves [3], Merrill [12] and Eaves and Saigal [4] proposed variations where one can initiate the computation at any point, and where the grid of search, δ, can be continuously refined to enable one to obtain a sequence of approximate fixed points which converges to a fixed point of f.

The algorithms of [4] and [12] can, in addition, be used to compute fixed points in problems where the region is not bounded. This feature results in the ability to compute fixed points of (1.1) under other hypothesis. One such example is the

Leray-Schauder Theorem [13, 6.3.3]. Let C be a bounded, open subset of R^n containing 0 and let f be a continuous mapping from the closure of C into R^n such that $f(x) \neq \lambda x$ whenever $\lambda > 1$ and x is in the boundary of C. Then f has a fixed point in the closure of C.

These later improvements in the algorithm are also based on the methods of complementary pivoting, and have a very appealing geometric interpretation. These algorithms can be viewed as starting with a simply constructed mapping f^0 and its fixed point x_0 in C, deforming f^t to $f^\infty = f$ as t goes from 0 to ∞, and following the path x_t of fixed points of f^t. Under rather general conditions on f (like the conditions of the Leray-Schauder theorem), the path x_t can be shown to converge to a fixed point of f. An extensive study of the path x_t thus generated has been made by Eaves [5]. In particular, a concept of index can be defined, and this index can be shown to be invariant along the path. See also [16].

In many applications including mathematical programming and economics, one needs to consider more general types of relations than functions. In particular, fixed point theorems dealing with point-to-set mappings, i.e. functions from R^n into subsets of R^n are needed. The most celebrated of these is

Kakutani's Theorem [7]. Let S be an n-dimensional simplex, and let S* be the class of all nonempty convex subsets of S. Let f be a mapping from S into S*, and let it be upper semicontinuous. Then, there is an x in S such that x is in f(x).

The algorithms developed in [4], [12] are designed to compute fixed points of point-to-set mappings, and are effective under conditions similar to those of the Leray-Schauder theorem.

Recent work of Saigal [15] has demonstrated that for a continuously differentiable function f whose derivative satisfies a Lipschitz condition, these algorithms can be made to converge quadratically to a fixed point, a desirable characteristic shared with the Newton's method, [13, 7.1]. In the same work, smooth problems of upto 80 dimensions are solved.

The use of these algorithms can be made even when the conditions which guarantee success cannot be verified. In [12], many nonlinear programming problems are successfully solved, though the conditions can only be verified for a few. In case the algorithm fails on a problem, very little can be concluded. Ideally, a conclusion that there is no fixed point would be valuable, but no specific results that do this exist. To date, many fairly large problems of upto 30 dimensions for point-to-set mappings, and 80 dimensions for smooth mappings have been successfully solved by these methods.

2. SOME STANDARD APPLICATIONS

Some applications of the fixed point computing methods include the economic equilibrium problem [18], nonlinear programming problem [12], nonlinear boundary value problem [1], solving systems of nonlinear equations [15], and finding roots of complex polynomials [9]. We will now discuss two of these applications.

The Nonlinear Programming Problem

The problem we consider is that of finding x to

$$\text{minimize} \quad \theta(x)$$

$$\text{subject to } g_i(x) \leq 0 \ , \quad i = 1,\ldots,m \ ,$$

where θ and g_i are real valued functions of n variables $x_1,\ldots,x_n$. In addition, we shall assume that each of these functions is convex, and that in case the set $G = \{x: \ g_i(x) \leq 0 \ i = 1,\ldots,m\}$ is nonempty, there is an x such that $g_i(x) < 0$ for each $i = 1,\ldots,m$.

For a convex function θ, a point x^* in R^n is called a subgradient at x if and only if

$$\theta(y) \geq \theta(x) + \langle x^*, y-x \rangle \quad \text{for all } y \text{ in } R^n.$$

where $\langle x,y \rangle = x_1y_1 + \ldots + x_ny_n$.
The subdifferential of θ at x is the set $\partial\theta(x)$ of all subgradients of θ at x. A fact about convex functions is that $\partial\theta(x) \neq \phi$ for each x. Also $\partial\theta(x)$ is closed and <u>convex</u> for each x.

Now, define the mapping

$$s(x) = \max_{1 \leq i \leq m} g_i(x)$$

and note that $s(x)$ is convex. Also, let

$$I(x) = \{i: \ s(x) = g_i(x)\} \neq \phi$$

for each x. We note that

$$\partial s(x) = \text{hull}\left\{ \bigcup_{i \in I(x)} \partial g_i(x) \right\}$$

where hull (C) is the smallest convex set containing C.

Now, consider the following mapping from R^n into convex subsets of R^n:

$$f(x) = \begin{cases} x - \partial\theta(x) & s(x) < 0 \\ x - \text{hull}\{\partial\theta(x) \cup \partial s(x)\} & s(x) = 0 \\ x - \partial s(x) & s(x) > 0 \end{cases}$$

It is readily confirmed that this mapping is upper semicontinuous. Now, let $\bar{x}$ be a fixed point of this mapping. We now show that $\bar{x}$ solves the nonlinear programming problem. Depending on the value of $s(x)$, we have the following three cases:

Case 2.1. $s(\bar{x}) < 0$. Then, $0 \in \partial\theta(\bar{x})$ and thus $\bar{x}$ is a global minimizer of θ, and is also in G, and thus solves the problem.

Case 2.2. $s(\bar{x}) > 0$. Then $0 \in \partial s(\bar{x})$, and $\bar{x}$ is a global minimizer of s, and since $s(\bar{x}) > 0$, the set G is empty.

Case 2.3. $s(\bar{x}) = 0$. Then $0 \in \text{hull}\{\partial\theta(\bar{x}) \cup \partial s(\bar{x})\}$, and thus there exist nonnegative numbers ρ_0, ρ_i $i \in I(\bar{x})$, and vectors $y_0 \in \partial\theta(\bar{x})$ and $y_i \in \partial g_i(\bar{x})$, $i \in I(\bar{x})$ such that

$$\rho_0 y_0 + \sum_{i \in I(\bar{x})} \rho_i y_i = 0$$

Now, if $\rho_0 = 0$, then since $\bar{x}$ is a global minimizer of s, $g_i(x) < 0$ for each i is impossible. Since, we have assumed the contrary, $\rho_0 \neq 0$, and we have $\bar{x}$ and $\frac{\rho_i}{\rho_0}$, $i \in I(x)$ satisfying the standard Karush-Kuhn-Tucker necessary and sufficient conditions for a solution to the problem.

Thus, solving a nonlinear programming problem can be reduced to finding a fixed point of the mapping f. Extensions to the general (non-convex case) appear in [12].

As an example, consider the problem

$$\text{minimize } x_1^2 + x_2^2 - 2x_1 - 3x_2$$

$$\text{subject to } x_1 + x_2 \leq 1.$$

Then, the mapping

$$f(x_1, x_2) = \begin{cases} (-x_1+2, -x_2+3) & x_1 + x_2 < 1 \\ \text{hull } \{(-x_1+2, -x_2+3), (x_1-1, x_2-1)\} & x_1 + x_2 = 1 \\ (x_1-1, x_2-1) & x_1 + x_2 > 1 \end{cases}$$

has, as a fixed point $\left(\frac{1}{4}, \frac{3}{4}\right)$, and note that this also solves the nonlinear programming problem.

Computing Economic Equilibrium Prices

We consider an economy in which n commodities are exchanged, a typical vector of commodities being denoted by $x = (x_1,\ldots,x_n)$. We also assume that there are m traders participating in this exchange, each endowed with a utility function $U_i(x_1,\ldots,x_n)$ $i = 1,\ldots,m$ on the space of commodities which specifies the individual's preferences; that is, the individuals choose a vector of commodities from those available to him, which maximizes his utility function. In addition, each trader has an initial vector of assets

$$\omega^i = \left(\omega_1^i,\ldots,\omega_n^i\right)$$

reflecting his ownership of commodities prior to trading. The problem we then consider is to find a price vector $\pi = (\pi_1,\ldots,\pi_n)$, where π_i is the price of the commodity i, in the economy such that trading at these prices would result in no shortages of goods in the economy.

Now, given an arbitrary set of prices $\pi = (\pi_1,\ldots,\pi_n)$ of the commodities in the economy, the wealth W^i of the i^{th} trader is obtained by assuming that he disposes off his entire holdings at these prices, so

$$W^i = \sum_{j=1}^{n} \pi_j \omega_j^i$$

Also, given the wealth of each trader in the economy, his demand for the commodities is determined (as stated before) by maximizing his utility function subject to the constraint that his expenditure must not exceed his wealth, that is, by solving:

$$\text{maximize} \quad U_i(x_1,\ldots,x_n)$$

$$\text{subject to} \quad \sum_{j=1}^{n} \pi_j x_j \leq W^i$$

$$x_j \geq 0, \quad j = 1,\ldots,n .$$

Thus, a vector of demands $D^i(\pi) = \left(D_1^i(\pi),\ldots,D_n^i(\pi)\right)$ which solves the above problem of the i^{th} trader would result.

As can be readily seen, these demands $D^i(\pi)$ as functions of π are homogeneous of degree 0. Thus, we can normalize the price vector to satisfy

$$\sum_{j=1}^{n} \pi_j = 1$$

$$\pi_j \geq 0, \quad j = 1,\ldots,n ,$$

that is, lie in an (n-1)-dimensional simplex, S.

As an example, if the utility function of a trader were

$$U(x_1,\ldots,x_n) = x_1^{a_1}x_2^{a_2}\ldots x_n^{a_n}$$

with $a_i \geq 0$ and $\sum_{i=1}^{n} a_i = 1$, then the demand vector would be

$$D(\pi) = \left(\frac{a_1 W}{\pi_1}, \frac{a_2 W}{\pi_2}, \ldots, \frac{a_n W}{\pi_n}\right),$$

where W is the wealth given in terms of the initial holding $(\omega_1,\ldots,\omega_n)$ by

$$W = \sum_{j=1}^{n} \pi_j \omega_j .$$

Now, summing the individual demands $D_j^i(\pi)$, we obtain the Market Demands $D(\pi) = (D_1(\pi),\ldots,D_n(\pi))$ where

$$D_j(\pi) = \sum_{i=1}^{m} D_j^i(\pi) .$$

For each commodity, define the <u>excess demand</u> function

$$g_j(\pi) = D_j(\pi) - \omega_j$$

where $\omega_j = \omega_j^1 + \ldots + \omega_j^m$, the total commodity j in the economy shared among the m traders.

The demand functions, in addition, satisfy another structural condition known as Walras Law, which states that

$$\pi_1 g_1(\pi) + \ldots + \pi_n g_n(\pi) = 0$$

for all price vectors π, whether they are in equilibrium or not.

An equilibrium price vector $\bar{\pi}$ is one for which excess demands are less than or equal to zero, or

$$g_i(\bar{\pi}) \leq 0 \quad i = 1,\ldots,n ,$$

thus at these prices, the total market demand can be satisfied by the initial stock of assets.

Now, let $k = (k_1,\ldots,k_n) > 0$ be fixed. Define

$$s_i(\pi) = \max\{-\pi_i, k_i g_i(\pi)\} \quad i = 1,\ldots,n$$

Clearly $s(\pi)$ is continuous when $g(\pi)$ is. Now, define the mapping

$$f(\pi) = \frac{\pi + s(\pi)}{\langle \pi + s(\pi), e \rangle}$$

where e is the n-vector of all 1's.

It can be readily confirmed that $\pi + s(\pi) \geq 0$. It also follows that $\langle \pi + s(\pi), e \rangle > 0$ for all $\pi \in S$, which then implies that f is a continuous mapping from S into S; and by Brouwer's theorem, it has a fixed point $\bar{\pi}$. We now show that this fixed point is an equilibrium price vector. Since

$$\bar{\pi} = \frac{\bar{\pi} + s(\bar{\pi})}{\langle \bar{\pi} + s(\bar{\pi}), e \rangle}$$

$$s(\bar{\pi}) = \lambda\bar{\pi} \quad \text{for some } \lambda \,,$$

or $$\langle s(\bar{\pi}),\ g(\bar{\pi})\rangle = \lambda \langle \bar{\pi},\ g(\bar{\pi})\rangle = 0$$

Also $s_i(\bar{\pi})\cdot g_i(\bar{\pi}) \geq 0$; and since $g_i(\bar{\pi}) > 0$ implies $s_i(\bar{\pi}) = k_i g_i(\bar{\pi}) > 0$, it must be that $g_i(\bar{\pi}) \leq 0$ for each $i = 1,\ldots,n$, and the result follows.

These equilibrium prices in such an economy are of interest since they reflect the relative scarcity of commodities in this economy.

3. THE BASICS OF THE ALGORITHM

We now illustrate the basics of the procedure by considering the problem of computing a fixed point of a continuous mapping f which maps the n-dimensional simplex

$$S = \left\{ x: \sum_{i=1}^{n+1} x_i = 1,\ x_i \geq 0 \right\} \subset R^{n+1}$$

into itself. The first step in the procedure is to subdivide the simplex S in such a way that each piece of the subdivision is an n-simplex, and such that any two n-simplexes that meet in this subdivision do so on a common face. A standard subdivision of S is obtained by choosing any integer $K \geq 1$, and considering all vectors in S of the form

$$\left(\frac{y_1}{K}, \ \ldots\ , \frac{y_{n+1}}{K} \right)$$

with $\sum_{i=1}^{n+1} y_i = K$ and y_i integer, as the vertices of the subdivision. An n-simplex of this subdivision is then uniquely obtained by a choice of a permutation π of $\{1,\ldots,n\}$ and a vertex v of this subdivision. The vertices $v^1,\ldots,v^{n+1}$ of the simplex (v,π) then are

$$\begin{aligned} v^1 &= v \\ v^{i+1} &= v^i + \frac{1}{K} Q_{\pi(i)} \qquad i = 1,\ldots,n \end{aligned} \tag{3.1}$$

where $\pi(i)$ is the i^{th} component of the permutation π and Q_i is the i^{th} column of the $(n+1)\times(n+1)$ matrix.

$$\begin{bmatrix} -1 & & & & & \\ 1 & -1 & & & & \\ & 1 & \ddots & & & \\ & & \ddots & \ddots & & \\ & & & \ddots & -1 & \\ & & & & 1 & \end{bmatrix}$$

It can be verified that the size of any such simplex, called the mesh of the subdivision, is no greater than $\sqrt{n}/K$. A standard subdivision of a 2-simplex with $K = 4$ is shown in Figure 3.1.

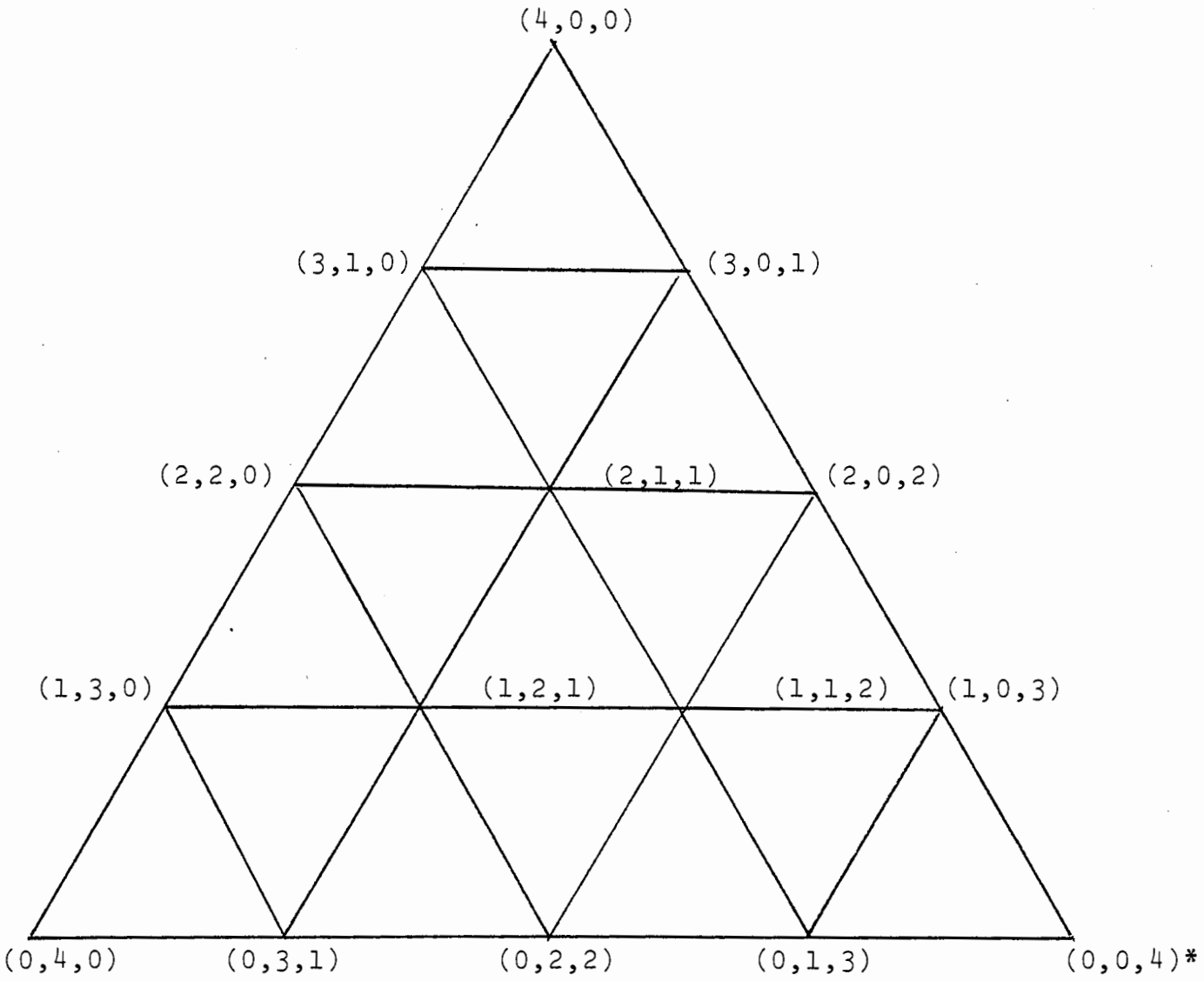
Given a subdivided simplex S and a continuous function f, one defines a labeling function ℓ, which associates a number from $\{1,\ldots,n+1\}$ to each of the vertices of the subdivision. This labeling function is generally

$$\ell(x) = j \quad \text{if} \quad f_j(x) \leq x_j \text{ and } x_j \neq 0 \qquad (3.2)$$

and is well defined, since

$$\sum_{j=1}^{n+1} f_j(x) = \sum_{j=1}^{n+1} x_j = 1$$

implies that $f_j(x) > x_j$ for all j with $x_j > 0$ is impossible.



A subdivided 2-simplex with K = 4.

Figure 3.1

* The coordinates of the vertices should be divided by 4.

As an example to demonstrate the labeling, consider the mapping f: $S \to S$ where $S = \{x: x_1+x_2 = 1, x_1 \geq 0, x_2 \geq 0\}$ and

$$f(x_1,x_2) = \frac{1}{\theta(x_1)} \begin{bmatrix} \frac{1}{2} + \frac{x_1^2}{2} - \frac{x_2^2}{3} \\ \frac{1}{2} - \frac{x_1^2}{3} + \frac{x_2^2}{2} \end{bmatrix}$$

where $\theta(x_1) = \frac{7}{6} - \frac{1}{3} x_1 + \frac{1}{3} x_1^2$. Also

$$f(1,0) = \left(\frac{6}{7} , \frac{1}{7}\right), \text{ the label } \ell(1,0) = 1 .$$

$$f(0,1) = \left(\frac{1}{7} , \frac{6}{7}\right), \text{ the label } \ell(0,1) = 2 .$$

On such an assignment, certain n-simplexes in the subdivision became completely labeled, that is, their vertices carry all the labels 1 through n + 1. Let σ be such a completely labeled simplex with vertices $v^1,\ldots,v^{n+1}$ with vertex v^i labeled i, for each $i = 1,\ldots,n+1$. Then it can be established that for any given $\varepsilon > 0$, if the size of the simplex σ is $\delta > 0$ (determined by the uniform continuity of f), any point x in σ is such that $||f(x) - x|| \leq n(\varepsilon+\delta)$. Thus, the task of computing approximate fixed points can be accomplished by searching, through the subdivision, for a completely labeled simplex. The algorithms are designed to precisely do this.

Figure 3.2 shows a subdivision of a 2-simplex with K = 7. It is in this setting that we shall describe the workings of the algorithm. Clearly, in view of (3.2), we need only specify the labeling of the vertices of the subdivision, and not the function which lead to such a labeling. Please note that the labeling rule implies a point x cannot be labeled i if $x_i = 0$. In the setting of the Sperner's lemma, such a labeling is called proper. The lemma then guarantees that there must be an odd number of completely labeled simplexes in the subdivision. As a by-product, we will also be proving this lemma.

Consider all line segments in the Figure 3.2 carrying both the labels 1 and 2, and all the completely labeled triangles. Now, place a cross in the middle of each such line and in the center of each such triangle. By joining these crosses by dashed lines whenever they belong to the same triangle in the subdivision, we can create at most four types of paths as shown in the Figure 3.2. Also, note that a path never returns to a triangle through which it has passed. These types can be enumerated as follows.

3.3 Paths starting at a line segment on the boundary, and ending at another line segment on the boundary (the path between A and B).

3.4 Paths starting inside a triangle of the subdivision and ending in a line segment on the boundary (the path between F and G).

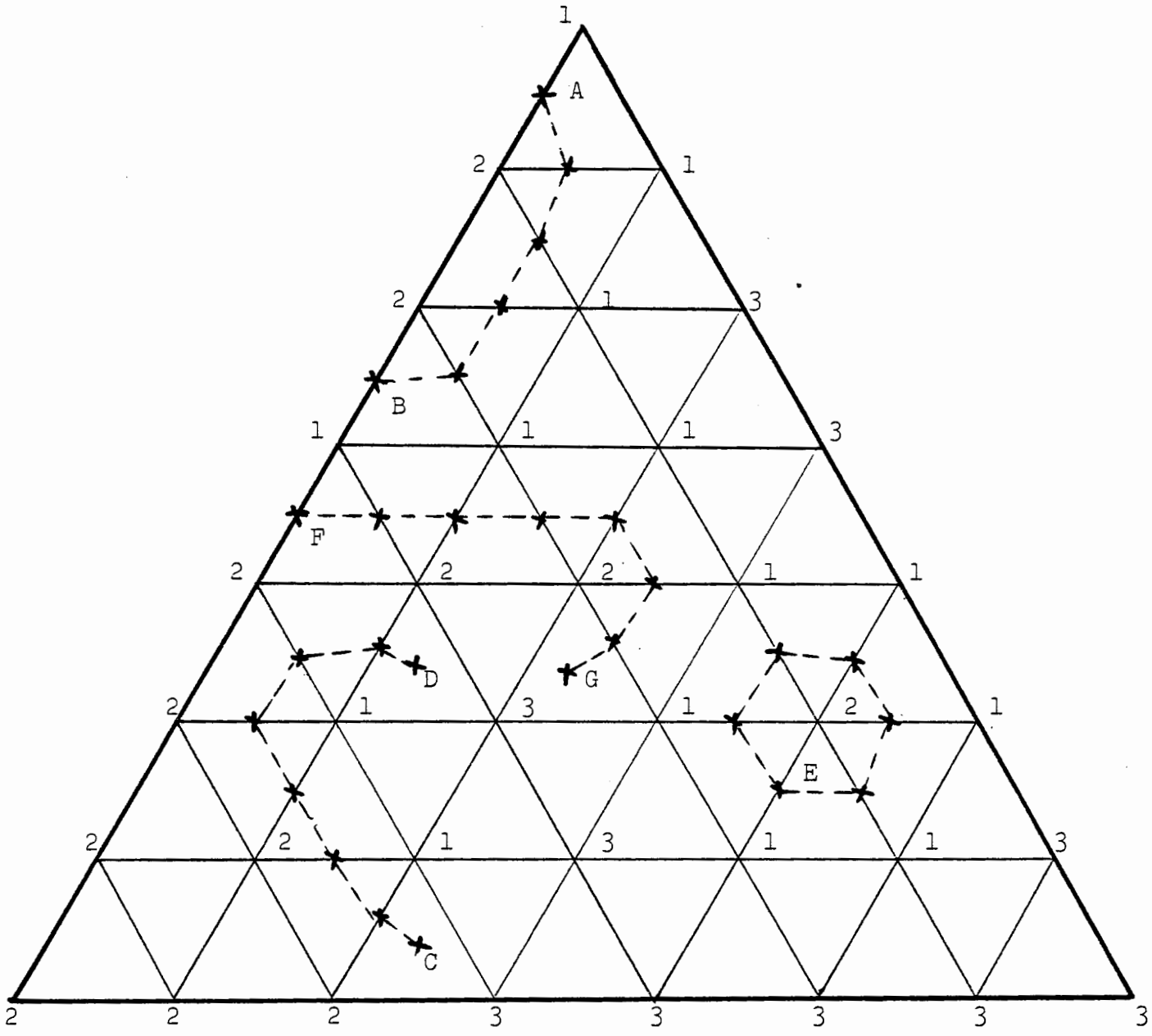


Figure 3.2

3.5 Paths starting inside a triangle and ending inside another triangle (the path between C and D).

3.6 Paths ending as a loop (E). (The algorithm never reaches these paths.)

The paths of type 3.3 and 3.4 suggest an algorithm for finding a completely labeled triangle. The basic elements of such an algorithm are the following:

Step 0: Find a line segment L on the boundary that has both the labels 1 and 2. There is only one triangle T of the subdivision on this line. Let the vertex of the triangle not on this line segment be v.

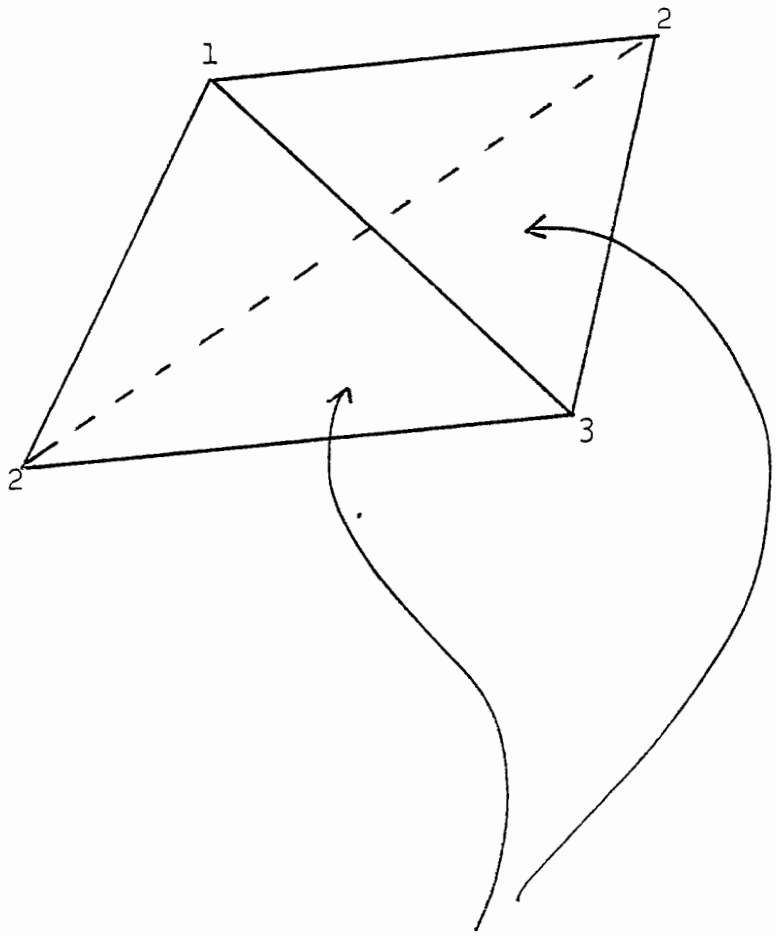
Step 1: If the label on v is 3, stop. Otherwise go to step 2.

Step 2: Since the label on v is either 1 or 2, one of the labels 1 or 2 is repeated in the triangle T. Hence, in T, there is exactly one more line segment which has both the labels 1 and 2. Let this line segment now be L.

Step 3: If L is on the boundary, stop. Otherwise, there is another triangle T on L (on the opposite side to the previous triangle). Make v the vertex on this triangle T not on L, and go to step 1.

The sequence of steps described in the above algorithm are finite since no triangle is considered more than once, and there are only a finite number of triangles in the subdivision. This is so since for a triangle to appear more than once in the procedure, it must have more than two line segments labeled 1 and 2, which is not the case (as seen in Step 2). Thus, after a finite search, the algorithm will stop at step 1 or step 3. Stopping in step 1 generates the path type 3.4, which then leads to a completely labeled triangle, while step 3 leads to a type 3.3 path, which has failed to produce a completely labeled triangle. The termination in step 3 is a real possibility, and in an implementable algorithm one guarantees that this will not happen by providing a unique starting line segment on the boundary.

The generalization of this algorithm to higher dimensions is now obvious. If we make the straightforward analogy between a triangle and an n-simplex, line segment and a (n-1)-dimensional face of the n-simplex, it can be readily seen that all the elements of the algorithm are maintained. Thus, the algorithm would start with a (n-1)-dimensional face in the boundary which has all the labels 1, 2 through n. If the label on the additional vertex on the unique n-simplex on this face is repeated, then there is <u>exactly one more</u> (n-1)-face which carries all the labels 1, 2 through n. (As was the case in 2 dimensions. This is shown in Figure 3.3 for 3 dimensions.) In addition, a subdivision in higher



These are the only 2-faces which carry all the labels 1 through 3.

Figure 3.3

dimensions also shares the property that each (n-1)-dimensional face on the boundary belongs to exactly one n-simplex, and otherwise to exactly two.

Thus, summarizing the basic steps in the development of an algorithm, one needs a procedure, called triangulation, for subdividing the space; a labeling method such that the distinguished simplexes called completely labeled can be related, in some manner, to fixed points of the mapping; under the labeling method, certain facets (faces of dimension 1 less than the dimension of a simplex of the subdivision) became distinguished, and any simplex containing these distinguished facets is either completely labeled (thus contains exactly one such facet) or contains exactly two such facets; and finally, to avoid termination at step 3, the boundary of the space under consideration must contain exactly one distinguished facet.

The computer implementation of such an algorithm involves the determination of the next simplex when a vertex with the repeated label has been dropped. For the subdivision procedure described in this section, this is readily obtained as follows:

Let the present simplex be (v,π), where $v = (v_1,\ldots,v_{n+1})$ and $\pi = (\pi_1,\ldots,\pi_{n+1})$, and its vertices (generated recursively by (3.1)) be $v^1,\ldots,v^{n+1}$. Also, assume that the vertex whose label is repeated is v^i. Then the new simplex is $(\bar{v},\bar{\pi})$ where $\bar{v}$ and $\bar{\pi}$ are obtained from Table 3.1.

	$\bar{v}$	$\bar{\pi}$
$i = 1$	$v + \frac{1}{k} Q_{\pi_1}$	$(\pi_2,\ldots,\pi_{n+1},\pi_1)$
$2 \leq i \leq n$	v	$(\pi_1,\ldots,\pi_{i+1},\pi_i,\ldots,\pi_{n+1})$
$i = n + 1$	$v - \frac{1}{k} Q_{\pi_{n+1}}$	$(\pi_{n+1},\pi_1,\ldots,\pi_n)$

TABLE 3.1

Now, if the vertices of the new simplex $(\bar{v},\bar{\pi})$ are $\bar{v}^1,\bar{v}^2,\ldots,\bar{v}^{n+1}$ the entering vertex is $\bar{v}^j$ where

$$j = \begin{cases} n+1 & i = 1 \\ i & 2 \leq i \leq n \\ 1 & i = n+1 \end{cases} \qquad (3.7)$$

The flow chart of the computer program implementing an algorithm on the subdivision of this section is given in Figure 3.4.

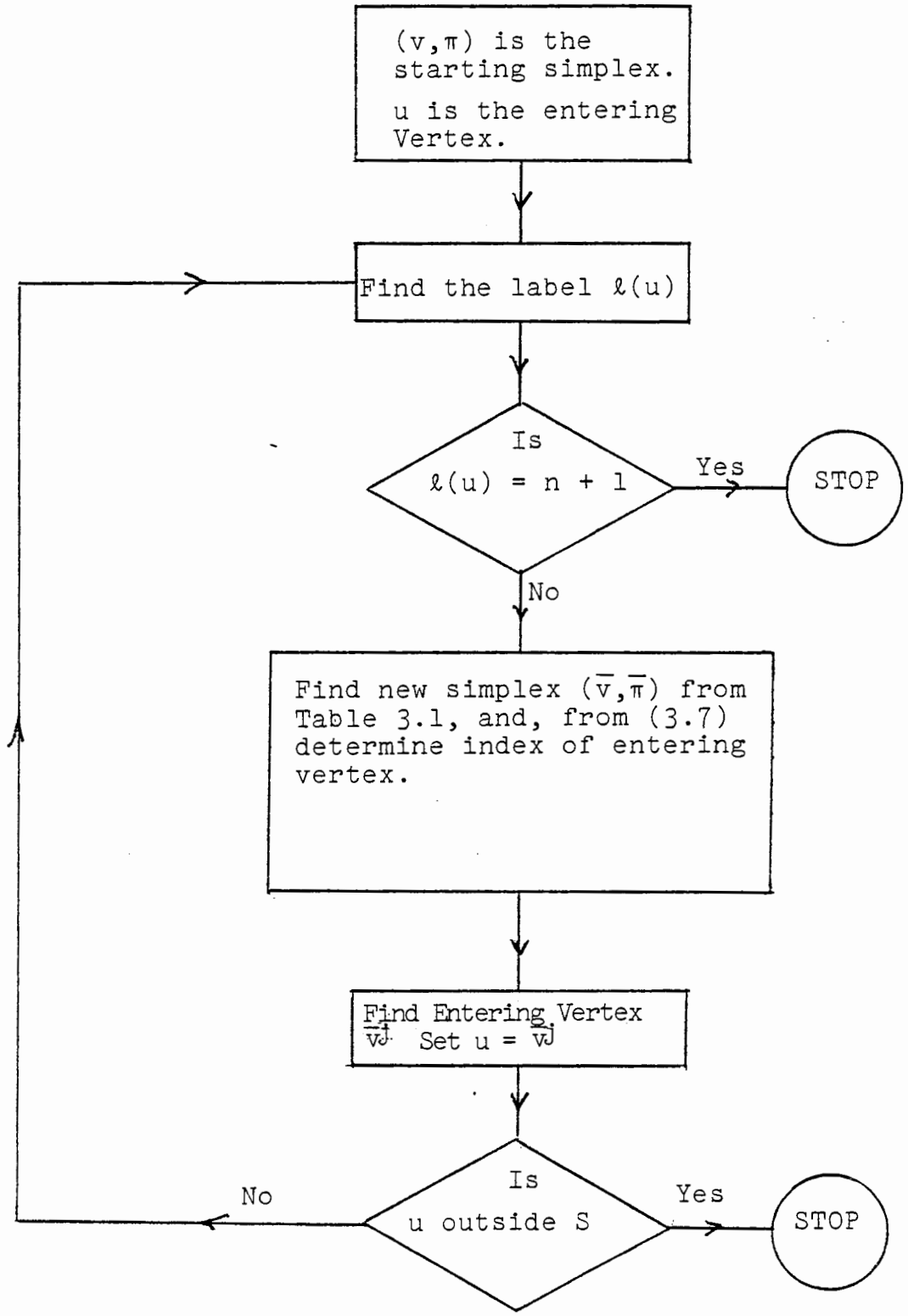

(v,π) is the starting simplex.
u is the entering Vertex.
Find the label ℓ(u)
Is ℓ(u) = n + 1
Yes
STOP
No
Find new simplex (v̄,π̄) from Table 3.1, and, from (3.7) determine index of entering vertex.
Find Entering Vertex v̄j. Set u = v̄j
Is u outside S
No
Yes
STOP

Figure 3.4

4. AN ALGORITHM

One of the major disadvantages of the algorithm developed in Section 3 is that one is restricted to initiate the search at the boundary of the space. R^n has no such boundary, and thus the ability to initiate the algorithm at any point is required. This is achieved by considering the space $R^n \times (0,D]$ for the search where D is some positive real number. This space now has the boundary $R^n \times \{0\} \cup R^n \times \{D\}$. The algorithm will be developed on a subdivision of this space such that a unique distinguished facet is available in $R^n \times \{D\}$ to initiate the algorithm. We shall now describe the various details of the method. Note that we are considering computing a fixed point of the mapping f from R^n into R^n.

The Labeling:

Let $r(x) = Ax - a$ be an affine mapping, with A a $n \times n$ nonsingular matrix. We will label a point (x,t) in $R^n \times (0,D]$ by the rule that

$$\ell(x,t) = \begin{cases} r(x) & \text{if } t = D \\ f(x) - x & \text{if } t < D \end{cases}$$

where $\ell(x,t)$ is the label on the point (x,t). As is evident, this labeling is quite different from the one used in Section 3.

Note that the simplexes in the subdivision of $R^n \times (0,D]$ are (n+1)-dimensional, and their facets n-dimensional.

We shall call a (n+1)-simplex σ with vertices $\left(v^1,t_1\right),\left(v^2,t_2\right),\ldots,\left(v^{n+2},t_{n+2}\right)$ distinguished if the system of equations

$$\sum_{i=1}^{n+2} \lambda_i \ell\left(v^i,t_i\right) = 0 \qquad (4.1)$$

$$\sum_{i=1}^{n+2} \lambda_i = 1 \qquad (4.2)$$

$$\lambda_i \geq 0,\ i = 1,\ldots,n+2 \qquad (4.3)$$

has a solution. This is a system with (n+2) variables and (n+1) equations, and in view of (4.2), is bounded. In case all solutions are <u>nondegenerate</u> (in the usual linear programming sense, i.e., each solution has at least n+1 <u>positive</u> variables), it can be shown that this system has <u>two</u> solutions with <u>exactly</u> n + 1 positive variables. These solutions correspond, in a unique way, with two facets of the simplex. We shall consider these two facets as the distinguished facets (as required for our labeling).

Let x^D be such that $Ax^D - a = 0$, and let $(v^1,D),(v^2,D),\ldots,(v^{n+1},D)$ be the vertices of a facet τ in $R^n\times\{D\}$. We note that this facet is distinguished if and only if (x^D,D) is in τ, and thus if we require (x^D,D) to lie in a unique facet of the subdivision, a unique starting facet on the boundary results.

We now confirm that distinguished facets under the labeling ℓ are related to fixed points of f. Let τ be a distinguished facet with vertices $\left(v^1,t_1\right),\ldots,\left(v^{n+1},t_{n+1}\right)$ with $t_i < D$, $i = 1,\ldots,n+1$. Also, let $\varepsilon > 0$ be arbitrary and given, and let $\delta > 0$ be determined by the uniform continuity of f, and let the size of τ be less than δ. Then, there is a point $\bar{x}$ in τ such that $||f(\bar{x}) - \bar{x}|| \leq \varepsilon$. To see this, note that since $t_i < D$, 4.1-4.3 reduces to

$$\sum_{i=1}^{n+1} \lambda_i \left(f(v^i) - v^i\right) = 0$$

$$\sum_{i=1}^{n+1} \lambda_i = 1$$

$$\lambda_i \geq 0, \quad i = 1,\ldots,n+1$$

And since τ is distinguished, the above system has a solution $(\bar{\lambda}_1,\ldots,\bar{\lambda}_{n+1})$. Now, define

$$\bar{x} = \sum_{i=1}^{n+1} \bar{\lambda}_i v^i$$

and note that

$$\sum_{i=1}^{n+1} \bar{\lambda}_i f(v^i) = \bar{x} \ .$$

Hence

$$f(\bar{x}) - \bar{x} = f(\bar{x}) - \sum_{i=0}^{n} \bar{\lambda}_i f(v^i)$$

and since $||\bar{x}-v^i|| \leq \delta$, we have the result.

Another disadvantage of the basic algorithm is that it works with a fixed mesh. The algorithm we develop here will also have the property that the mesh of a facet τ in $R^n \times \left\{\frac{D}{2^k}\right\}$ goes to zero as k approaches ∞, and thus starting with a unique distinguished facet τ_0 in $R^n \times \{D\}$, one will find, progressively, distinguished facets τ_k in $R^n \times \left\{\frac{D}{2^k}\right\}$ for k approaching ∞. We now describe the triangulation procedure for obtaining the subdivision of $R^n \times (0,D]$ which in addition, has the above property of the facets.

The Triangulation J3

The triangulation procedure called J3 in the literature, for obtaining the subdivision of $R^n \times (0,D]$ was developed by Todd [19], and is related to the triangulation K3 of [4] on which the original algorithm was developed and described in 1971. We now describe this triangulation.

Given an arbitrary positive real number D, the vertices J_3^0 of this triangulation are all points v in R^{n+1} such that $v_{n+1} = D.2^{-k}$ for some integer k = 0,1,2,... and for each i = 1,...,n, v_i/v_{n+1} are integers. In addition, a subset $\bar{J}_3^0$ of vertices in J_3^0 are called central vertices if for each $v \in \bar{J}_3^0$ and i = 1,...,n, v_i/v_{n+1} is an odd integer.

A vertex v in J_3^0 is said to have depth k if $v_{n+1} = D \cdot 2^{-k}$. Note that to each central vertex v of depth $k \geq 1$, there is a unique <u>nearest</u> central vertex of depth k - 1 obtained as follows:

Define $\nu(v) = (\nu_1(v), \nu_2(v), \ldots, \nu_{n+1}(v))$ where

$$\nu_i(v) = \begin{cases} -1 & \text{if } v_i/v_{n+1} \text{ is } 1 \bmod 4 \\ +1 & \text{if } v_i/v_{n+1} \text{ is } 3 \bmod 4 \end{cases}$$

Then the nearest central vertex to v is

$$y(v) = v - v_{n+1} \cdot \nu(v) \ .$$

Now, each simplex σ in J3 has a unique representation by a triplet (v,π,s) where v is a central vertex, π a permutation of $\{1,\ldots,n+1\}$ and $s = (s_1,\ldots,s_{n+1})$ with $s_i \in \{-1,+1\}$, $i = 1,\ldots,n+1$. If $\nu = \nu(v)$, the vertices of this simplex are generated as follows (where $\pi(j) = n+1$):

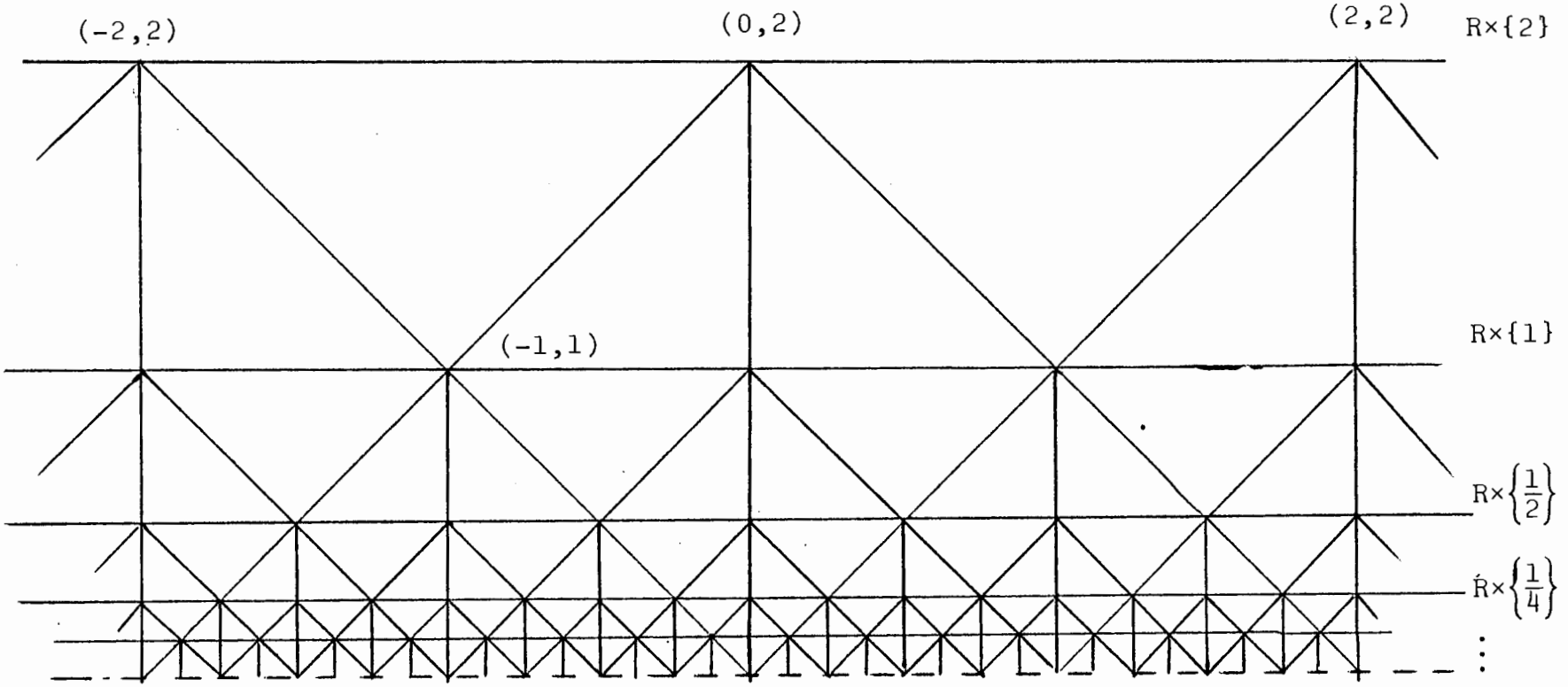
$$\begin{aligned} v^1 &= v \\ v^{i+1} &= v^i + v_{n+1} s_{\pi(i)} u_{\pi(i)} \qquad i = 1,\ldots,j-1 \\ v^{j+1} &= v^j - v_{n+1} \sum_{\ell=j+1}^{n+1} \nu_{\pi(\ell)} u_{\pi(\ell)} + v_{n+1} u_{n+1} \\ v^{k+1} &= v^k + 2 v_{n+1} \nu_{\pi(k)} u_{\pi(k)} \qquad j+1 \leq k \leq n+1 \end{aligned} \tag{4.4}$$

The simplex σ is thus the convex hull of the vertices $v^1,\ldots,v^{n+2}$. That J3 is indeed a triangulation is proved in [19]. The triangulation J3 of Rx(0,2] is shown in Figure 4.1.

Now, let (v,π,s) be a simplex in J3, and let $\pi(j) = n + 1$. Also, let $v^1,\ldots,v^{n+2}$ be the vertices of this simplex generated by the recursive relations (4.4), and assume that the vertex which has a repeated label (and is hence dropped) is v^i. The new simplex $(\bar{v},\bar{\pi},\bar{s})$ is then obtained by the rules given in Table 4.1. (The recursive relations can be simplified by defining a n-dimension vector b such that $b_i \in \{-1,+1\}$ and substituting $b_{\pi(i)}$ for $s_{\pi(i)}$ whenever $1 \leq i \leq j - 1$, and for $\nu_{\pi(i)}$ for $j + 1 \leq i \leq n + 1$. This, in effect, combines ν and s into one vector. The Table 4.1 updates b rather than ν and s separately).

Pivoting

Implementation of the method of this section requires the determination of the vertex that has a "repeated label". This determination was simple in the basic algorithm, since it was easily recognizable. In the method under consideration, this determination requires a pivot operation, identical to one in the simplex method of the linear programming theory. This is now discussed in some detail below.



Triangulation J3 of R×(0,2]

Figure 4.1

		$\bar{v}$	$\bar{\pi}$	$\bar{b}$
$i = 1$	$j = 1$	$v - v_{n+1}(-1,b)$	$(\pi(2),\ldots,\pi(n+1),\pi(1))$	b
	$j > 1$	$v + 2v_{n+1}b_{\pi(1)}u^{\pi(1)}$ *	π	$b - 2b_{\pi(1)}u^{\pi(1)}$ *
$2 \leq i \leq j - 1$		v	$(\pi(1),\ldots,\pi(i+1),\pi(i),\ldots,\pi(n+1))$	b
$i = j$	$b_{\pi(j-1)}=v_{\pi(j-1)}$	v	$(\pi(1),\ldots,\pi(j),\pi(j-1),\ldots,\pi(n-1))$	b
	$b_{\pi(j-1)}=-v_{\pi(j-1)}$	v	$(\pi(1),\ldots,\pi(j-2),\pi(j),\ldots,\pi(n+1),\pi(j-1))$	$b - 2b_{\pi(j-1)}u^{\pi(j-1)}$
$j + 1 \leq i \leq n + 1$		v	$(\pi(1),\ldots,\pi(i+1),\pi(i),\ldots,\pi(n+1))$	b
$i = n + 2$	$j = n + 1$	$v + \frac{1}{2} v_{n+1}(-1,b)$	$(\pi(n+1),\pi(1),\ldots,\pi(n))$	b
	$j < n + 1$	v	$(\pi(1),\ldots,\pi(j-1),\pi(n+1),\pi(j),\ldots,\pi(n))$	$b - 2b_{\pi(n+1)}u^{\pi(n+1)}$

TABLE 4.1

* u^i has been generically used for the i^{th} unit vector of the correct dimension.

As we have seen, the method is initiated with a distinguished facet τ on the boundary $R^n \times \{D\}$ of the space. Let the vertices of this facet be $(v^1,D),(v^2,D),\ldots,(v^{n+1},D)$. Define the matrix

$$B = \begin{bmatrix} r(v^1) & r(v^2) & \cdots & r(v^{n+1}) \\ 1 & 1 & & 1 \end{bmatrix}$$

and note that this is a $(n+1)\times(n+1)$ nonsingular matrix. Also, since the facet τ is distinguished, there is a solution $\bar{\lambda}$ to the system

$$B\lambda = b$$

$$\lambda \geq 0$$

where $b = \begin{bmatrix} \vec{0} \\ 1 \end{bmatrix}$ and $\vec{0}$ is the n-dimensional null vector.

Now, let σ be the unique simplex on this facet, let (v,t) be the vertex of this simplex not in τ and let $\ell(v,t) \equiv c$ be the label on this vertex. The "repeated label" is now determined by that column of B which is replaced by $\begin{pmatrix} c \\ 1 \end{pmatrix}$ such that for the resulting matrix $\bar{B}$ the system

$$\bar{B}\lambda = b$$

$$\lambda \geq 0$$

has a solution. This determination is carried out in precisely the same manner as in the simplex method with the lexicographic pivot rule to guarantee that a distinguished simplex will have exactly two distinguished facets. The lexicographic pivot also eliminates the need for such a nondegeneracy assumption.

The Geometry of the Algorithm

The algorithm just described has a very appealing geometric interpretation. We shall now describe this geometry.

Define a piecewise linear mapping H from $R^n \times (0,D]$ to R^n such that on any vertex (x,t) of the triangulation J3

$$H(x,t) = \ell(x,t)$$

and is linear on each simplex σ of the triangulation, that is, if the vertices of σ are $\left(v^1,t_1\right),\ldots,\left(v^{n+2},t_{n+2}\right)$ and (x,t) in σ is such that

$$(x,t) = \sum_{i=1}^{n+2} \lambda_i \left(v^i,t_i\right) \qquad (4.5)$$

then

$$H(x,t) = \sum_{i=1}^{n+2} \lambda_i H\left(v^i,t_i\right) . \qquad (4.6)$$

As can be readily confirmed, $H(x,D) = r(x)$ and $H\left(\cdot, \frac{D}{2^k}\right)$ is a piecewise linear approximation to $f(x) - x$ on a triangulation of $R^n \times \left\{\frac{D}{2^k}\right\}$. Also, as the size of the facets in this space approaches 0 as k approaches ∞, this approximation approaches $f(x) - x$ uniformly.

Now, let τ be a distinguished facet with vertices $\left(v^1, t_1\right), \ldots, \left(v^{n+1}, t_{n+1}\right)$. Then, there is a point (x,t) in τ such that $H(x,t) = 0$. This follows since (4.1-3) holds for distinguished facets, and (4.5-6) then gives the result. Also, since H is linear on any simplex σ, if σ is distinguished, there is a <u>line segment</u> $\left(x_h(\sigma), t_h(\sigma)\right)$ $h \in [0,1]$, in σ, such that $\left(x_0(\sigma), t_0(\sigma)\right)$ and $\left(x_1(\sigma), t_1(\sigma)\right)$ lie in the boundary of σ and $H\left(x_h(\sigma), t_h(\sigma)\right) = 0$ for each $h \in [0,1]$.

Since $r(x)$ is one-to-one and linear, $H(x,D) = 0$ has a unique solution (x^D, D). Thus, starting with the point (x^D, D) in $R^n \times \{D\}$, the algorithm generates a <u>piecewise linear</u> path (x_h, t_h), h going from 0 to infinity such that

$$(x_0, t_0) = (x^D, D)$$

and

$$H(x_h, t_h) = 0$$

for each h. (Such a path is obtained by "joining" the linear pieces in each distinguished simplex generated by the algorithm.) In addition, if $t_h \to 0$, since $H(\cdot,t_h)$ approaches $f(x) - x$, all cluster points of x_h are fixed points of f. (In [4], a modification of the labeling is given which insures that in every case, t_h will approach 0.)

Conditions for Success

We now state some conditions on the mapping f under which the piecewise linear path generated by the algorithm will converge to a fixed point of f. We will assume that f is uniformly continuous and r(x) is one-to-one affine mapping with $r(x_0) = 0$. Our first result is

Theorem 4.1. Let there exist an open, bounded set C containing x_0, and a $\delta > 0$ such that any simplex of size less than δ which intersects the boundary of C is not distinguished. Then, there is a $D > 0$ such that the path starting at x_0 never leaves C, and thus converges to a fixed point of f.

Proof: Let $D = \frac{\delta}{\sqrt{n+1}}$. The algorithm generates simplexes with diameter less than δ, and starts inside C. Since none of these simplexes intersect the boundary of C, they all lie inside C. Also, as for each $t > 0$, there are a finite number of simplexes of J3 inside $C \times [t,D]$, in the path (x_h,t_h) thus generated t_h must approach 0 as h approaches infinity. Hence x_h has a cluster point, and we have the result.

The next result can be considered as a generalization of the Leray-Schauder theorem. Here $h(x) = f(x) - x$.

Theorem 4.2. Let C be an open bounded set containing x_0 such that $h(x) + \rho r(x) \neq 0$ for all $\rho \geq 0$ and x in the boundary of C. Then, there is a $D > 0$ such that the algorithm will compute a fixed point of f.

Proof: Assume the contrary. Then, for each D_k, $k = 1,2,\ldots$ such that $D_k \to 0$ as $k \to \infty$, there is a distinguished simplex σ^k which intersects the boundary of C. Let x^k be a point in this intersection. Since x^k lies in a compact region, there is a subsequence $x^{k'}$ which converges to a point $\bar{x}$, on the boundary of C. In addition $\sigma^{k'}$ also converge to $\bar{x}$, (since $D_{k'} \to 0$). Since σ^k is distinguished, there is an m_k such that if the vertices of σ^k are $v_1^k,\ldots,v_{n+2}^k$

$$\sum_{i=1}^{m_k} \lambda_i^k h\left(v_i^k\right) + \sum_{i=m_k+1}^{n+2} \lambda_i^k r\left(v_i^k\right) = 0$$

where λ_i^k are nonnegative and sum to 1. Since $m_{k'}$ lie in the set $\{1,\ldots,n+2\}$ and $\lambda_i^{k'}$, $i = 1,\ldots,n+2$ lie in the compact set $[0,1]$, there is a subsequence k'' of k' such that $m_{k''} = m$ and $\lambda_i^{k''} \to \lambda_i$ for each $i = 1,\ldots,n+2$. Also, on this subsequence $v_i^{k''} \to \bar{x}$. Thus, using the continuity of h and r, we obtain

$$h(\bar{x}) \cdot \sum_{i=1}^{m} \lambda_i + r(\bar{x}) \cdot \sum_{i=m+1}^{n+2} \lambda_i = 0 .$$

Since $r(x) \neq 0$ for all x in the boundary of C, and $\bar{x}$ is such a point, we conclude that $h(\bar{x}) + \rho r(\bar{x}) = 0$ with

$$\rho = \frac{\sum_{i=m+1}^{n+2} \lambda_i}{\sum_{i=1}^{m} \lambda_i} > 0.$$

and we have a contradiction.

To see how the Leray-Schauder theorem follows from Theorem 4.2, define $r(x) = -x$ and $h(x) = f(x) - x$. Then $f(x) \neq \lambda x$ for all $\lambda \geq 1$ if and only if $h(x) + \rho r(x) \neq 0$ for all $\rho \geq 0$. Thus, starting the algorithm at $x_0 = 0$, we have the proof.

5. COMPUTATIONAL CONSIDERATIONS

In some uses of these algorithms, specially in economics, fixed points are known to exist, and a reasonable choice of the initial affine mapping r can be made so that the algorithms will compute a fixed point. Saigal [16] has demonstrated that even when the mapping is affine (i.e., $f(x) = Ax-a$) and there is a unique fixed point, a proper choice of r has to be made, otherwise the algorithm fails to compute it. Thus, since in a practical problem choosing r and D to satisfy the convergence conditions of the previous section may be very time consuming, a reasonable approach would be to make an arbitrary choice and apply the algorithm. If failure occurs, hopefully enough will be learnt about the mapping to modify the choice of r. At present there is no theory which lays the guidelines for such a modification, but the results of [16] are potentially useful. Unfortunately, if the failure is due to the nonexistence of a fixed point such an approach could be very frustrating.

A reasonable approach for detecting failure may be to keep track of the entering vertex v, and terminate when $\sum_{i=1}^{n} |v_i| > B$, for some predefined positive number B. In practical problems, B can be generally defined.

Some attention has been paid to the study of growth of computational effort as a function of the dimension of the problem. By making reasonable hypothesis on the behavior of the algorithm, Saigal [14] and Todd [20] have

associated measures on triangulations which would give an indication of this growth. Theoretically one shows that these measures, which count the number of simplexes the algorithm would generate to pass through a unit region of the space, grow approximately as the square of the dimension n. Since it takes $O(n^2)$ multiplications to pass through each simplex, it is expected that computational work would grow as $O(n^4)$. This seems to be supported by a growing body of computational experience.

BIBLIOGRAPHY

[1] Allgower, E. L. and M. M. Jeppson, "The Approximation of Solutions of Non-linear Elliptic Boundary Value Problems with Several Solutions," Springer Lecture Notes 333, (1973), 1-20.

[2] Brouwer, L. E. J., "Über eineindentige, stetige Transformationen von Flächen in Sich," Mathematische Annalen, Volume 67, (1910), 176-180.

[3] Eaves, B. C., "Homotopies for Computation of Fixed Points," Mathematical Programming, Volume 3, (1972), 1-22.

[4] Eaves, B. C. and R. Saigal, "Homotopies for Computation of Fixed Points on Unbounded Regions," Mathematical Programming, Volume 3, (1972), 225-237.

[5] Eaves, B. C., "A Short Course in Solving Equations with PL Homotopies," Proceedings of the ninth SIAM-AMS symposium in Applied Mathematics, Eds. R. W. Cottle and C. E. Lemke, to appear.

[6] Hansen, T., "On the Approximation of a Competitive Equilibrium," Ph.D Thesis, Yale University, (1968).

[7] Kakutani, S., "A Generalization of the Brouwer's Fixed Point Theorem," Duke Mathematical Journal, Volume 8, (1941), 457-459.

[8] Kuhn, H. W., "Simplicial Approximation of Fixed Points," Proceedings of the National Academy of Science, U.S.A., Volume 61, (1968), 1238-1242.

[9] Kuhn, H. W., "A New Proof of the Fundamental Theorem of Algebra," Mathematical Programming Study I, (1974), 148-158.

[10] Lemke, C. E. and J. T. Howson, Jr., "Equilibrium Points of Bimatrix Games, SIAM Journal on Applied Mathematics, Volume 12, (1964), 413-423.

[11] Lemke, C. E., "Bimatrix Equilibrium Points and Mathematical Programming," Management Science, Volume II, (1965), 681-689.

[12] Merrill, O. H., "Applications and Extensions of an Algorithm That Computes Fixed Points of Certain Upper Semi-Continuous Point to Set Mappings," Ph.D Dissertation, University of Michigan, Ann Arbor, Michigan, (1972), 228 pp.

[13] Ortega, J. M. and W. C. Rheinboldt, Iterative Solutions of Nonlinear Equations in Several Variables, Academic Press, New York - London, (1970).

[14] Saigal, R., "Investigations into the Efficiency of Fixed Point Algorithms," Fixed Points - Algorithms and Applications, Ed. S. Karamardian, to appear in 1976.

[15] Saigal, R., "On the convergence rate of Algorithms for solving equations that are based on methods of Complementary Pivoting," submitted to Mathematics of Operations Research.

[16] Saigal, R., "On Paths Generated by Fixed Point Algorithms," to appear in Mathematics of Operations Research.

[17] Scarf, H., "The Approximation of Fixed Points of a Continuous Mapping," SIAM Journal on Applied Mathematics, Volume 15, (1967), 1328-1343.

[18] Scarf, H., The Computation of Economic Equilibrium, in colaboration with T. Hansen, Yale University Press, New Haven, (1973), 249 pp.

[19] Todd, M. J., "Union Jack Triangulations," Fixed Points: Algorithms and Applications, Ed. S. Karamardian, to appear in 1976.

[20] Todd, M. J., "On Triangulations for Computing Fixed Points," Department of Operation Research, Cornell University, Ithaca, N.Y., (1974), 33 pp. To appear in Mathematical Programming in 1976.