Mathematical Programming 27 (1983) 107–120
North-Holland

DISTRIBUTED ASYNCHRONOUS COMPUTATION OF FIXED POINTS*

Dimitri P. BERTSEKAS

Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139, U.S.A.

Received 2 October 1981
Revised manuscript received 20 August 1982

We present an algorithmic model for distributed computation of fixed points whereby several processors participate simultaneously in the calculations while exchanging information via communication links. We place essentially no assumptions on the ordering of computation and communication between processors thereby allowing for completely uncoordinated execution. We provide a general convergence theorem for algorithms of this type, and demonstrate its applicability to several classes of problems including the calculation of fixed points of contraction and monotone mappings arising in linear and nonlinear systems of equations, optimization problems, shortest path problems, and dynamic programming.

Key words: Fixed Points, Distributed Algorithm, Optimization, Shortest Path, Dynamic Programming.

1. Introduction

There is presently a great deal of interest in distributed implementations of various iterative algorithms whereby the computational load is shared by several processors while coordination is maintained by information exchange via communication links. In most of the work done in this area the starting point is some iterative algorithm which is guaranteed to converge to the correct solution under the usual circumstances of centralized computation in a single processor. The computational load of the typical iteration is then divided in some way between the available processors, and it is assumed that the processors exchange all necessary information regarding the outcomes of the current iteration before a new iteration can begin.

The mode of operation described above may be termed synchronous in the sense that each processor must complete its assigned portion of an iteration and communicate the results to every other processor before a new iteration can begin. This assumption certainly enchances the orderly operation of the algorithm and greatly simplifies the convergence analysis. On the other hand synchronous distributed algorithms also have some obvious implementation disadvantages such as the need for an algorithm initiation and iteration synch-

*This research was contacted at the M.I.T. Laboratory for Information and Decision Systems with partial support provided by the Defense Advanced Research Projects Agency under Grant No. ONR-N00014-75-C-1183.

ronization protocol. Furthermore the speed of computation is limited to that of the slowest processor. It is thus interesting to consider algorithms that can tolerate a more flexible ordering of computation and communication between processors. Such algorithms have so far found applications in computer communication networks such as the ARPANET [9] where processor failures are common and it is quite complicated to maintain synchronization between the nodes of the entire network as they execute real-time network functions such as the routing algorithm. They could also find application in special purpose multiprocessors of the type that are currently being implemented by several research groups.

Given a distributed algorithm it is natural to try to determine the minimum degree of synchronization between computation and communication that is necessary in order for the algorithm to work correctly. In this paper we consider an extreme model of asynchronous distributed algorithms whereby computation and communication is performed at the various processors completely independently of the progress in other processors. Perhaps somewhat surprisingly we find that even under these potentially chaotic circumstances of uncoordinated computation it is possible to solve correctly broad and significant classes of fixed point problems by means of the natural distributed version of the successive approximation method. A general convergence theorem is developed for this purpose which delineates circumstances under which convergence is guaranteed. The theorem is then applied to broad classes of fixed point problems involving contraction and monotone mappings.

The nature of the distributed algorithm and the convergence result of this paper can be illustrated by considering the following example involving iterations of the Gauss–Seidel type for solving systems of nonlinear equations.

Fixed points of mappings on R^n: Consider the problem of finding an n-dimensional vector x^* which is a fixed point of a mapping $f : R^n \to R^n$, i.e.

$$x^* = f(x^*).$$

Let x_i and f_i, $i = 1, \ldots, n$ denote the coordinates of x and f respectively, and consider iterations of the form

$$x_1 \leftarrow f_1(x_1, x_2, \ldots, x_n), \tag{1.1}$$

$$x_2 \leftarrow f_2(x_1, x_2, \ldots, x_n), \tag{1.2}$$

$$\vdots \qquad\qquad \vdots$$

$$x_n \leftarrow f_n(x_1, x_2, \ldots, x_n). \tag{1.3}$$

In order to give precise meaning to iterations (1.1)–(1.n) we must specify the order in which they are executed and the rule by which the values of the coordinates $x_1, \ldots, x_n$ are chosen in the right side of each iteration. There are several ways of doing this that lead to well-known algorithms. Assume for example that all computations are done at a single central processor and at some

time instant k the vector x has the value $x^k = (x_1^k, \dots, x_n^k)$. A possible iteration (the classical successive approximation method) is

$$x^{k+1} = f(x^k). \tag{2}$$

It corresponds to all iterations (1.1)–(1.n) being carried out 'simultaneously', i.e. without substituting in the right side of (1.1)–(1.n) the most recently computed values of the coordinates of x. This can be contrasted with the following Gauss–Seidel type iteration

$$x_1^{k+1} = f_1(x_1^k, x_2^k, \dots, x_n^k), \tag{3.1}$$

$$x_2^{k+1} = f_2(x_1^{k+1}, x_2^k, \dots, x_n^k), \tag{3.2}$$

$$\vdots \qquad\qquad \vdots$$

$$x_n^{k+1} = f_n(x_1^{k+1}, x_2^{k+1}, \dots, x_{n-1}^{k+1}, x_n^k) \tag{3.n}$$

where the most recently computed values of the coordinates are being used.

Iterations (1.1)–(1.n) lend themselves well to distributed computation by n processors each assigned the responsibility of executing only one of these iterations and using communication links to exchange with other processors the most recent result of their respective computations. Iteration (2) corresponds to a synchronous model whereby each processor i executes its assigned iteration (1.i) and then communicates the updated value x_i^{k+1} to all other processors. After all updated values are exchanged the process is repeated. Iteration (3.1)–(3.n) corresponds to a sequential model whereby processor 1 executes (1.1), transmits x_1^{k+1} to all other processors, then processor 2 executes (1.2), transmits x_2^{k+1} to all other processors and so on. One of the sharpest general convergence results available for iterations (2) and (3.1)–(3.n) is that a sufficient condition for convergence to the unique fixed point of f is that f be a *P-contraction mapping* [11, p. 433]. By this we mean that there exists an $n \times n$ matrix P with nonnegative elements, with spectral radius strictly less than unity, and such that

$$|f(x) - f(y)| \leq P|x - y| \quad \forall\, x, y \in R^n \tag{4}$$

where for any vector $z = (z_1, \dots, z_n)$ we denote by $|z|$ the column vector with coordinates the absolute values $|z_1|, \dots, |z_n|$, and the inequality in (4) is meant to hold separately for each coordinate.

There is a variety of ways of executing iterations (1.1)–(1.n) other than (2) or (3.1)–(3.n). For example the order in which the iterations are executed may change as time progresses. An even bolder step is to assume that not only the order of the iterations may be arbitrarily changed, but also *the values of the coordinates in the right side of the iterations may be arbitrarily out-of-date.* For example (1.1) may be executed on the basis of the values of $x_1, \dots, x_n$ most recently computed but (1.2) may be executed on the basis of values of $x_1, x_3, \dots, x_n$ computed, say, one hundred computation 'cycles' ago. A surprising fact—a consequence of the general convergence result of this paper—is that

even under these extreme circumstances the assumption (4) is still sufficient to guarantee convergence of the resulting algorithm.[1]

In order to prove results such as the one briefly described above it is necessary to introduce a precise distributed computation model since the traditional concept of an iteration does not fully capture the essense of distributed algorithms of the type we are interested in. This is done in the next section. In Section 3 we develop our main convergence result while in Section 4 we analyze some important special cases.

2. A model for distributed asynchronous fixed point algorithms

The fixed point problem considered in this paper is defined in terms of a set X and a function $f : X \to X$. We wish to find an element $x^* \in X$ such that

$$x^* = f(x^*). \tag{5}$$

Each $x \in X$ is defined in terms of 'coordinates' x_i, $i \in I$ where I is a possibly infinite index set, i.e. we have $x = \{x_i \mid i \in I\}$. Each coordinate x_i is either a real number or $\pm\infty$. Similarly f is defined in terms of its coordinate functions f_i, $i \in I$ where $f_i(x) \in [-\infty, +\infty]$ for all $x \in X$ and $i \in I$. Therefore (5) can be equivalently written as

$$x_i^* = f_i(x^*) \quad \forall\, i \in I. \tag{6}$$

If I has n elements, $I = \{1, 2, \ldots, n\}$, and x_i is a real number for each i, then the problem is simply to find a fixed point of an n-dimensional mapping on a subset X of the Euclidean space R^n—the example considered in the previous section. Evidently all problems of solving a system of n nonlinear equations with n unknowns, as well as many problems of n-dimensional unconstrained and constrained optimization can be posed in this manner. The case where x_i can take the values $+\infty$ and $-\infty$ is genuinely interesting as it arises in dynamic programming problems (see [4, 2, Chapters 6 and 7]). Despite the fact that in any practical implementation of the algorithm of this paper the index set I must be finite and the coordinates x_i must be real numbers (indeed bounded precision rationals), it is useful for analytical purposes to consider the problem in the more general framework described above.

An interesting example of problem (5) is the shortest path problem for which the algorithm of this paper bears close relation with a routing algorithm ori-

[1] While this paper was under review the author became aware of considerable related work of Chazan and Miranker [5], Miellou [10], and Baudet [1] on asynchronous relaxation methods. The result just stated is proved in essence by these authors by different methods. The computation model considered by these authors is similar but is less general and differs in essential details from ours. The convergence result of this paper is much more general—for example it applies to dynamic programming algorithms.

ginally implemented in the ARPANET and subsequently in several other computer networks [9].

Shortest path problem: Let (I, A) be a directed graph where $I = (1, \dots, n\}$ denotes the set of nodes and A denotes the set of arcs. Let $N(i)$ denote the downstream neighbors of node i, i.e., the set of nodes j for which (i, j) is an arc. Assume that each arc (i, j) is assigned a positive scalar a_{ij} referred to as its length. Assume also that there is a directed path to node 1 from every other node. Then it is known [7, 12] that the shortest path distances x_i^* to node 1 from all other nodes i solve uniquely the equations

$$x_i^* = \min_{j \in N(i)} \{a_{ij} + x_j^*\} \quad \forall\, i \neq 1,$$

$$x_1^* = 0.$$

If we make the identifications

$$f_i(x) = \begin{cases} \min_{j \in N(i)} \{a_{ij} + x_j\} & \text{if } i \neq 1, \\ 0 & \text{if } i = 1, \end{cases}$$

$$X = \{x \mid x_1 = 0,\ x_i \in [0, \infty],\ i = 2, \dots, n\},$$

then we find that the fixed point problem (6) reduces to the shortest path problem.

Actually the problem above is representative of a broad class of dynamic programming problems which can be viewed as special cases of the fixed point problem (6) and can be correctly solved by using the distributed algorithm of this paper (see [4]).

Our algorithmic model can be described in terms of a collection of n computation centers (or processors) referred to as *nodes* and denoted $1, 2, \dots, n$. The index set I is partitioned into n disjoint sets denoted $I_1, \dots, I_n$, i.e.

$$I = \bigcup_{j=1}^{n} I_j, \quad I_j \cap I_m = \emptyset, \quad \text{if } j \neq m.$$

Each node j is assigned the responsibility of computing the coordinates x_i^* of a fixed point x^* for all $i \in I_j$.

At each time instant, node j can be in one of three possible states *compute*, *transmit*, or *idle*. In the compute state node j computes a new estimate x_i for all $i \in I_j$. In the transmit state node j communicates the estimate obtained from its own latest computation to one or more nodes $m (m \neq j)$. In the idle state node j does nothing related to the solution of the problem. It is assumed that a node can receive a transmission from other nodes simultaneously with computing or transmitting, but this is not a real restriction since, if needed, a time period in a separate receive state can be lumped into a time period in the idle state.

We assume that computation and transmission for each node takes place in uninterupted time intervals $[t_1, t_2]$ with $t_1 < t_2$, but do not exclude the possibility

that a node may be simultaneously transmitting to more than one node nor do we assume that the transmission intervals to these nodes have the same origin and/or termination. We also make no assumptions on the length, timing and sequencing of computation and transmission intervals other than the following:

Assumption A. For every node j and time $t \geq 0$ there exists a time $t' > t$ such that $[t, t']$ contains at least one computation interval for j and at least one transmission interval from j to each node $m \neq j$.

Assumption A is very natural. It states in essence that no node 'drops out of the algorithm' permanently—perhaps due to a hardware failure. Without this assumption there is hardly anything we can hope to prove.

Each node j has a buffer B_{jm} for each $m \neq j$ where it stores the latest transmission from m, as well as a fubber B_{jj} where it stores its own estimate of the coordinates x_i of a solution for all $i \in I_j$. The contents for each buffer B_{jm} at time t are denoted $x^t(j, m)$. Thus $x^t(j, m)$ is, for every t, j and m a vector of coordinate estimates $\{x_i \mid i \in I_m\}$ available at node j at time t. It is important to realize in what follows that *the buffer contents* $x^t(j, m)$ *and* $x^t(j', m)$ *at two different nodes* j *and* j' *need not coincide at all times. If* $j \neq m$ *and* $j' \neq m$ *the buffer contents* $x^t(j, m)$ *and* $x^t(j', m)$ *need not coincide at any time* t. The vector of all buffer contents of node j is denoted $x^t(j)$, i.e.,

$$x^t(j) = \{x^t(j, m) \mid m = 1, \ldots, n\}.$$

The coordinates of $x^t(j)$ are denoted $x_i^t(j)$, $i \in I$, and the coordinates of $x^t(j, m)$ are denoted $x_i^t(j, m)$, $i \in I_m$.

The rules according to which the buffer contents $x^t(j, m)$ are updated are as follows:

(1) If $[t_1, t_2]$ is a transmission interval from node m to node j, the contents of the buffer B_{mm} at time t_1 are transmitted and entered in the buffer B_{jm} at time t_2, i.e.

$$x^{t_1}(m, m) = x^{t_2}(j, m). \tag{7}$$

(2) If $[t_1, t_2]$ is a computation interval for node j, the contents of the buffer B_{jj} at time t_2 are replaced by $f_i[x^{t_1}(j)]$, $i \in I_j$, i.e.

$$x_i^{t_2}(j) = f_i[x^{t_1}(j)] \quad \forall\, i \in I_j. \tag{8}$$

(3) The contents of a buffer B_{jj} can change only at the end of a computation interval for node j. The contents of a buffer B_{jm}, $j \neq m$ can change only at the end of a transmission interval from m to j.

Our objective is to derive conditions under which

$$\lim_{t \to \infty} x_i^t(j) = x_i^* \quad \forall\, i \in I,\ j = 1, \ldots, n, \tag{9}$$

where $x^* \in X$ is a fixed point of f. This is the subject of the next section.

3. A general convergence theorem

In our effort to develop a general convergence result for the distributed algorithmic model of the previous section we draw motivation from existing convergence theories for (centralized) iterative algorithms. There are several theories of this type (Zangwill [16], Luenberger [8], Daniel [6], Ortega and Rheinboldt [11]—the most general are due to Poljak [14] and Polak [13]). All these theories have their origin in Lyapunov's stability theory for differential and difference equations. The main idea is to consider a generalized distance function (or Lyapunov function) of the typical iterate to the solution set. In optimization methods the objective function is often suitable for this purpose while in equation solving methods a norm of the difference between the current iterate and the solution is usually employed. The idea is typically to show at each iteration the value of the distance function is reduced and reaches its minimum value in the limit.

Our result is based on a similar idea. However, instead of working with a generalized distance function we prefer to work (essentially) with the level sets of such a function.

We formulate the following assumption under which we will subsequently prove convergence of the type indicated in (9). In what follows $\bar{X}$ denotes the set of *all* vectors $x = \{x_i \mid x_i \in [-\infty, \infty],\ i \in I\}$, $X \times X$ denotes the Cartesian product of X with itself, and $\Pi_{j=1}^{n} X$ denotes the Cartesian product of X with itself n times.

Assumption B. There exists a sequence X^k of subsets of X with the following properties:

(a) If $\{x^k\}$ is a sequence in X such that if $x^k \in X^k$ for all k, then

$$\lim_{k \to \infty} x_i^k = x_i^* \quad \forall\, i \in I \tag{10}$$

where $x^* \in X$ is some fixed point of f.

(b) For all $k = 0, 1, \ldots$ and $j = 1, \ldots, n$

$$x \in X^k \Rightarrow f(x; j) \in X^k \tag{11}$$

where $f(\cdot\,; j) : X \to \bar{X}$ is the mapping defined by

$$f_i(x; j) = \begin{cases} x_i & \text{if } i \notin I_j, \\ f_i(x) & \text{if } i \in I_j. \end{cases} \tag{12}$$

(c) For all $k = 0, 1, \ldots$ and $j = 1, \ldots, n$

$$x \in X^k,\ x' \in X^k \Rightarrow C(x, x'; j) \in X^k \tag{13}$$

where $C(\cdot, \cdot; j): X \times X \to \bar{X}$ is the mapping defined by

$$C_i(x, x'; j) = \begin{cases} x_i & \text{if } x \notin I_j, \\ x_i' & \text{if } i \in I_j. \end{cases} \tag{14}$$

(d) For all $k = 0, 1, \ldots$

$$x^1 \in X^k, x^2 \in X^k, \ldots, x^n \in X^k \Rightarrow F(x^1, x^2, \ldots, x^n) \in X^{k+1}, \tag{15}$$

where $F: \Pi_{j=1}^n X \to \bar{X}$ is the mapping defined by

$$F_i(x^1, x^2, \ldots, x^n) = f_i(x^j) \quad \forall\, i \in I_j,\ j = 1, \ldots, n. \tag{16}$$

Assumption B seems rather complicated so it may be worth providing some motivation for introducing it. Property (a) specifies how the sets X^k should relate to a solution x^*. Property (d) guarantees that if the functions in the buffers of all nodes $i = 1, \ldots, n$ belong to X^k and a computation phase is carried out simultaneously at all nodes followed by a communication phase from every node to every other node, then the resulting function in the buffer of each node (which will be the same for all nodes), will belong to X^{k+1}. Properties (a) and (d) alone guarantee that the algorithm will converge to a correct solution if executed in a synchronous manner, i.e., a simultaneous computation phase at all nodes is followed by a simultaneous communication phase from each node to all other nodes and the process is repeated. Property (b) involves the mapping $f(\cdot; j)$ which is related to a computation phase at node j (compare (8) with (12)), while property (c) involves the mapping $C(\cdot, \cdot; j)$ which is related to a communication phase from node j to some other node (compare (7) with (13), (14)). Basically properties (b) and (c) guarantee that the sets X^k are closed with respect to individual node computation and communication. By this we mean that if all buffer contents are within X^k, then after a single node computation or communication all buffer contents will remain in X^k. The following proposition asserts that when properties (b) and (c) hold in addition to (a) and (d), then the algorithm converges to the correct solution when operated in a totally uncoordinated manner.

Proposition. *Let Assumptions* A *and* B *hold, and assume that the initial buffer contents* $x^0(j)$ *at each node* $j = 1, \ldots, n$ *belong to* x^0. *Then*

$$\lim_{t \to \infty} x_i^t(j) = x_i^* \quad \forall\, i \in I,\ j = 1, \ldots, n \tag{17}$$

where $x^* \in X$ *is a fixed point of* f *and* $x_i^t(j)$ *is the* ith *coordinate of the buffer content vector* $x^t(j)$ *of node* j *at time* t.

Proof. We will show that for every $k = 0, 1, \ldots$ and $t \geq 0$ the condition

$$x^t(j) \in X^k \quad \forall\, j = 1, \ldots, n \tag{18}$$

implies that there exists a time $t_1 > t$ such that

$$x^{t'}(j) \in X^k \quad \forall\, t' \geq t,\ j = 1, \ldots, n, \tag{19}$$

$$x^{t'}(j) \in X^{k+1} \quad \forall\, t' \geq t_1,\ j = 1, \ldots, n. \tag{20}$$

In view of condition (a) of Assumption B, this will suffice to prove the proposition.

Assume that (18) holds for some $k = 0, 1, \ldots$ and $t \geq 0$. Then (19) clearly holds since, for $t' \geq t$, the buffer content $x^{t'}(j)$ of node j at t' is obtained from the buffer contents $x^{t}(m)$ of all nodes $m = 1, \ldots, n$ at t via operations that (according to conditions (b) and (c) of Assumption B) preserve membership in X^k.

By assumption A there exists a scalar $\delta_1 > 0$ such that $[t, t + \delta_1]$ contains at least one computation interval for each node $j = 1, \ldots, n$. Therefore, using (8), we have that for each $t' \geq t + \delta_1$.

$$x_i^{t'}(j) = f_i[x^{\bar{t}}(j)] \quad \forall\, i \in I_j,\ j = 1, \ldots, n \tag{21}$$

where $x^{\bar{t}}(j)$ is the buffer content of node j at some time $\bar{t} \in [t, t + \delta_1]$ ($\bar{t}$ depends on j), and by (19)

$$x^{\bar{t}}(j) \in X^k \quad \forall\, j = 1, \ldots, n.$$

Using again Assumption A we have that there exists a scalar $\delta_2 > 0$ such that $[t + \delta_1, t + \delta_1 + \delta_2]$ contains at least one communication interval from every node to every other node. It follows that, for every $t' \geq t + \delta_1 + \delta_2$, each buffer B_{jm} contains a vector $x^{\tilde{t}}(j, m)$ such that (cf. (7), (21))

$$x_i^{\tilde{t}}(j, m) = x_i^{\tilde{t}}(m, m) = f_i[x^{\bar{t}}(m)] \quad \forall\, i \in I_m,\ j, m = 1, \ldots, n, \tag{22}$$

where $x^{\tilde{t}}(m, m)$ is the content of buffer B_{mm} at node m at some time $\tilde{t} \in [t + \delta_1, t + \delta_1 + \delta_2]$ and $x^{\bar{t}}(m)$ is the buffer content of node m at some time $\bar{t} \in [t, \tilde{t}]$. (Again here the times $\tilde{t}$ and $\bar{t}$ depend on j and m.)

Let $t_1 = t + \delta_1 + \delta_2$. By using (22) and (19) we can assert that for each $t' \geq t_1$ and $j = 1, \ldots, n$ there exist vectors $\bar{x}^j \in X^k$, $j = 1, \ldots, n$ such that

$$x_i^{t'}(j) = f_i(\bar{x}^j) \quad \forall\, i \in I_j,\ j = 1, \ldots, n,$$

It follows from condition (d) of Assumption B (cf. (15), (16)) that

$$x^{t'}(j) \in X^{k+1} \quad \forall\, t' \geq t_1,\ j = 1, \ldots, n$$

which is (20). This completes the proof of the proposition.

Note that (18) and (20) can form the basis for an estimate of the rate of convergence of the algorithm. For example if there exists an index $\bar{k}$ such that $X^k = X^{\bar{k}} = \{x^*\}$ for all $k \geq \bar{k}$ (i.e. after some index the sets X^k contain only one element—a fixed point $x^* \in X$), then it follows from (18)–(20) that the distributed algorithm converges to the correct solution in a finite amount of time.

This argument can, for example, be used to establish finite time convergence for the distributed algorithm as applied to the shortest path problem of Section 2.

4. Special cases

In this section we verify that Assumption B of the previous section is satisfied for some important classes of problems.

4.1. *Contraction mappings with respect to sup-norms*

Let $\tilde{X}$ be the vector space of all $x = \{x_i \mid x_i \in (-\infty, \infty), i \in I\}$ which are bounded in the sense that there exists $M > 0$ such that $|x_i| \leq M$ for all $i \in I$. Consider a norm on $\tilde{X}$ of the form

$$\|x\| = \sup_{i \in I} \alpha_i |x_i| \tag{23}$$

where $\{\alpha_i \mid i \in I\}$ is a set of scalars such that for some $\bar{\alpha} > 0$ and $\underline{\alpha} > 0$ we have

$$\underline{\alpha} \leq \alpha_i \leq \bar{\alpha} \quad \forall\, i \in I.$$

Assume that the set X either equals $\tilde{X}$ or is a closed sphere centered at a fixed point x^* of f. Assume further that f is a contraction mapping on X with respect to the norm (23) in the sense that, for some $p < 1$ we have

$$\|f(x) - f(y)\| \leq p\|x - y\| \quad \forall\, x, y \in X.$$

Then, because $\tilde{X}$ is a complete space and X is a closed subset of $\tilde{X}$, x^* is the unique fixed point of f in X (cf. [15]).

For $q > 0$ define

$$X^k = \{x \in \tilde{X} \mid \|x - x^*\| \leq p^k q\}, \quad k = 0, 1, \ldots .$$

It is evident that if $X^0 \subset X$, then the sequence $\{X^k\}$ satisfies conditions (a)–(d) of Assumption B.

We note that the use of a sup-norm such as (23) is essential in order for Assumption B to hold. If f is a contraction mapping with respect to some other type of norm, Assumption B need not be satisfied.

4.2. *P-contraction mappings*

Let $I = \{1, 2, \ldots, n\}$ and assume that X is a subset of R^n. Suppose that f is a P-contraction mapping, i.e. satisfies the condition

$$|f(x) - f(y)| \leq P|x - y| \quad \forall\, x, y \in R^n, \tag{24}$$

where P is an $n \times n$ matrix with nonnegative elements and spectral radius strictly less than unity, and for any $z = (z_1, z_2, \ldots, z_n)$ we denote by $|z|$ the column

vector with coordinates $|z_1|, |z_2|, \ldots, |z_n|$. Condition (24) holds in particular if P is a stochastic matrix (all elements of P are nonnegative and the sum of the elements of each row of P is less than or equal to unity) and $\lim_{k\to\infty} P^k = 0$. Fixed point problems involving P-contraction mappings arise in dynamic programming [2, p. 374], and solution of systems of nonlinear equations [11, Section 13.1].

It has been shown in [1, p. 231] that if f is a P-contraction, then it is a contraction mapping with respect to some norm of the form (23). We are therefore reduced to the case examined earlier.

4.3. *Monotone mappings*

Assume that f has the monotonicity property

$$x \in X,\ x' \in X,\ x_i \le x'_i,\ \forall\, i \in I \Rightarrow f_i(x) \le f_i(x'), \quad \forall\, i \in I. \tag{25}$$

Denote by f^k the composition of f with itself k times and assume that there exist two elements $\underline{x}$ and $\bar{x}$ of X such that

$$\{x \mid \underline{x}_i \le x_i \le \bar{x}_i,\ \forall\, i \in I\} \subset X \tag{26}$$

and for all $k = 0, 1, \ldots$

$$f_i^k(\underline{x}) \le f_i^{k+1}(\underline{x}) \le f_i^{k+1}(\bar{x}) \le f_i^{k-1}(\bar{x}) \quad \forall\, i \in I \tag{27}$$

and

$$\lim_{k\to\infty} f_i^k(\underline{x}) = \lim_{k\to\infty} f_i^k(\bar{x}) = x_i^* \quad \forall\, i \in I \tag{28}$$

where $x^* \in X$ is a fixed point of f.

As an example consider the shortest path problem in Section 2, and the function

$$\underline{x}_i = 0 \quad \forall\, i = 1, \ldots, n,$$

$$\bar{x}_i = \begin{cases} \infty & \text{if } i \neq 1, \\ 0 & \text{if } i = 1. \end{cases}$$

It is easily verified that the corresponding function f satisfies (25) and that $\underline{x}$, $\bar{x}$ as defined above satisfy (26), (27), (28).

Define now for $k = 0, 1, \ldots$

$$X^k = \{x \mid f_i^k(\underline{x}) \le x_i \le f_i^k(\bar{x}),\ \forall\, i \in I\}.$$

Then it is easily seen that the sequence $\{X^k\}$ satisfies conditions (a)–(d) of Assumption B.

Fixed point problems involving monotone mappings satisfying (25) arise in dynamic programming [2, 3, 4] and solution of systems of nonlinear equations [11, Section 13.2].

4.4. *Unconstrained optimization*

Consider the problem

$$\begin{aligned} &\text{minimize} \quad g(x), \\ &\text{subject to} \quad x \in R^n \end{aligned} \tag{29}$$

where $g : R^n \to R$ is a twice continuously differentiable convex function, with Hessian matrix $\nabla^2 g(x)$ which is positive definite for all x.

The mapping that corresponds to Newton's method is given by

$$f(x) = x - [\nabla^2 g(x)]^{-1} \nabla g(x) \tag{30}$$

where $\nabla g(x)$ denotes the gradient of g at x. Under the assumptions made earlier a vector x^* satisfying $\nabla g(x^*) = 0$ is the unique globally optimal solution of problem (29) and also the unique fixed point of the mapping f of (30). Suppose there exists such a vector x^*. Then it is a simple matter to verify that there exists an open sphere centered at x^* such that the mapping f of (30) is a contraction mapping in X with respect to the norm $\|x\| = \max_i |x_i|$. Therefore the distributed version of Newton's method is convergent if the starting buffer contents are sufficiently near x^*. A similar fact can be shown if the inverse Hessian $[\nabla^2 g(x)]^{-1}$ in (30) is replaced by a matrix $H(x)$ such that the difference $H(x) - [\nabla^2 g(x)]^{-1}$ has sufficiently small norm uniformly within X.

Consider next the mapping corresponding to the ordinary gradient method

$$f(x) = x - \alpha \nabla g(x) \tag{31}$$

where α is a positive scalar stepsize. Again if x^* is the unique optimal solution of problem (29), then x^* is the unique fixed point of f as given by (31). The Jacobian matrix of f is given by

$$\partial f(x) = I - \alpha \nabla^2 g(x) \tag{32}$$

where I is the $n \times n$ identity matrix. Using the mean value theorem we have for all $x, y R^n$

$$f_i(x) - f_i(y) = \sum_{j=1}^{n} \frac{\partial f_i(x^i)}{\partial x_j} (x_j - y_j) \quad \forall\, i = 1, \ldots, n \tag{33}$$

where x^i is a vector lying on the line segment joining x and y. From (33) we obtain

$$|f_i(x) - f_i(y)| \leq \sum_{j=1}^{n} \left| \frac{\partial f_i(x^i)}{\partial x_j} \right| |x_j - y_j|. \tag{34}$$

Denote by $|f(x) - f(y)|$ and $|x - y|$ the column vectors with coordinates $|f_i(x) - f_i(y)|$ and $|x_i - y_i|$ respectively. Assume that the stepsize α in (31) satisfies

$$\alpha \frac{\partial^2 g(x^i)}{(\partial x_i)^2} < 1 \quad \forall\, i = 1, 2, \ldots, n. \tag{35}$$

Then, with the aid of (32), we can write (34) as

$$|f(x) - f(y)| \leq F|x - y| \tag{36}$$

where F is the $n \times n$ matrix given by

$$F = I - \alpha G \tag{37}$$

and G is given by

$$G = \begin{bmatrix} \left|\frac{\partial^2 g}{(\partial x_1)^2}\right|, & -\left|\frac{\partial^2 g}{\partial x_1 \partial x_2}\right|, \cdots, & -\left|\frac{\partial^2 g}{\partial x_1 \partial x_n}\right| \\ \vdots & & \\ -\left|\frac{\partial^2 g}{\partial x_n \partial x_1}\right|, & -\left|\frac{\partial^2 g}{\partial x_n \partial x_2}\right|, \cdots, & \left|\frac{\partial^2 g}{(\partial x_n)^2}\right| \end{bmatrix} \tag{38}$$

and the derivatives in the ith row of the matrix above are evaluated at x^i.

It is now seen easily from (36) and (37) that f will be a P-contraction mapping within an open sphere centered at x^* provided the following two conditions hold:

(a) The matrix G^* is positive definite where G^* is given by (38) with all partial derivatives evaluated at x^*.

(b) The stepsize α is sufficiently small so that (35) holds and the matrix $1 - \alpha G^*$ (cf. (37)) is positive definite. Equivalently α should be smaller than the inverses of the largest eigenvalue and the largest diagonal element of G^*.

If the two conditions above are satisfied, then the distributed gradient algorithm based on the mapping f of (31) is convergent to x^* provided all buffer contents are sufficiently close to x^*.

Unfortunately it is not true that the matrix G^* is always positive definite and indeed examples can be constructed where the distributed gradient method can fail to converge to the optimal solution x^* regardless of the choice of the stepsize α. Despite this fact we believe that the distributed gradient method is an interesting algorithm. We will show in a forthcoming publication that it has satisfactory convergence properties provided we impose certain mild restrictions on the relative timing of computations and communications in place of Assumption A.

5. Conclusions

The analysis of this paper shows that broad classes of fixed point problems can be solved by distributed algorithms that operate under very weak restrictions on the timing and ordering of processor computation and communication phases. It is also interesting that the initial processor buffer contents need not be identical and can vary within a broad range. This means that for problems that are being solved continuously in real time it is not necessary to reset the initial

conditions and resynchronize the algorithm each time the problem data changes. As a result the potential for tracking slow changes in the solution function is improved and algorithmic implementation is greatly simplified.

References

[1] G.M. Baudet, "Asynchronous iterative methods for multiprocessors", *Journal of the ACM* 2 (1978) 226–244.
[2] D.P. Bertsekas, *Dynamic programming and stochastic control* (Academic Press, New York, 1976).
[3] D.P. Bertsekas and S.E. Shreve, *Stochastic optimal control: The discrete time case* (Academic Press, New York, 1978).
[4] D.P. Bertsekas "Distributed dynamic programming", *IEEE Transactions on Automatic Control* AC-27 (1982) 610–616.
[5] D. Chazan and W. Miranker, "Chaotic relaxation", *Linear Algebra and Its Applications* 2 (1969) 199–222.
[6] J.W. Daniel, *The approximate minimization of functionals* (Prentice Hall, Englewood Cliffs, NJ, 1971).
[7] E.L. Lawler, *Combinatorial optimization: Networks and matroids* (Holt, Rinehart, and Winston, New York, 1976).
[8] D.G. Luenberger, *Introduction to linear and nonlinear programming* (Addison-Wesley, Reading, MA, 1973).
[9] J. McQuillan, G. Falk and I. Richer, "A review of the development and performance of the ARPANET routing algorithm", *IEEE Transactions on Communications* COM-26 (1978) 1802–1811.
[10] J. C. Miellou, "Itérations chaotiques a retards, études de la convergence dans le case d'espaces partiellment ordonnés", *Comptes Rendus de l'Académie des Sciences, Paris, Série A* 278 (1974) 957–960.
[11] J.M. Ortega and W.C. Rheinboldt, *Iterative solution of nonlinear equations in several variables* (Academic Press, New York, 1970).
[12] C.H. Papadimitriou and K. Steiglitz, *Combinatorial optimization: Algorithms and complexity* (Prentice Hall, Englewood Cliffs, NJ, 1982).
[13] E. Polak, *Computational methods in optimization: A unified approach* (Academic Press, New York, 1971).
[14] B.J. Poljak, "Convergence and convergence rate of iterative stochastic algorithms", *Automation and Remote Control* 12 (1982) 83–94.
[15] H.L. Royden, *Real analysis* (MacMillan, New York, 1963).
[16] W.I. Zangwill, *Nonlinear programming* (Prentice Hall, Englewood Cliffs, NJ, 1969).