A METHOD OF TERMINATING ASYNCHRONOUS ITERATIVE ALGORITHMS ON MESSAGE PASSING SYSTEMS

Didier EL BAZ

LAAS du CNRS, 7, avenue du Colonel Roche,

31077 Toulouse Cedex, France

A method of terminating of asynchronous iterative algorithms on message passing architectures is presented. Asynchronous algorithms are slightly modified. The method is original, general, and requires few extra communications. A formal proof of validity of the proposed termination procedure is given.

KEY WORDS: distributed termination, asynchronous iterative algorithms, fixed point problems, message passing architectures.

C.R. CATEGORIES: D1.3, G1.0.

1. INTRODUCTION

In this paper we study the solution of fixed point problems via asynchronous iterative methods whereby computations are carried out by several processors without any order nor synchronization. We concentrate on the termination of asynchronous iterations on message passing architectures. In this case the termination problem presents many difficulties since processors possess only local information, there is no global clock, and the communication delays may be unpredictable. This topic is related to numerical analysis (termination must occur when the iteration vector is sufficiently close to a solution of the numerical problem) and computer science (a special procedure must be designed for the observation of the state of the processors, the detection of the termination, and finally, the termination of the different processes). We note that this problem is relevant to the termination detection of distributed processes (see [1] to [3]). However asynchronous algorithms iterate in general indefinitely and computation processes are never idle.

In this paper we propose an original and general method for terminating asynchronous iterative algorithms which requires few additional communications. This method can be applied with success to all asynchronous iterative algorithms satisfying the conditions of the Asynchronous Convergence Theorem of Bertsekas (see [4] and [5, p. 431]).

In Section two we present asynchronous iterations and some basic termination methods. Section three deals with a new distributed termination procedure.

2. TERMINATION OF ASYNCHRONOUS ITERATIVE ALGORITHMS

We consider in this paper the solution of fixed point problems, $p = F(p)$, where F is a mapping from R^n onto itself. We concentrate on asynchronous iterative algorithms which are sequences $\{p(k)\}$ of vectors of R^n defined as follows (see [5, subsection 6.1]).

2.1. Asynchronous Iterative Algorithms

DEFINITION 2.1. We assume that there is a set of times $T = \{0, 1, 2, ...\}$ at which one or more components of vector p are updated by some processor. Let T^i be the subset of times at which component p_i is updated. For all $i \in \{1, ..., n\}$, the sequences $\{p_i(k)\}$ are defined recursively by:

$$p_i(k+1) = F_i(p_1(\tau_1^i(k)), ..., p_n(\tau_n^i(k))), \forall k \in T^i, p_i(k+1) = p_i(k), \forall k \notin T^i, \tag{2.1}$$

where F_i is the i-th component of the mapping F and for all $i \in \{1, ..., n\}$:

(a) the set T^i is infinite,

(b) $0 \leq \tau_j^i(k) \leq k, \forall j \in \{1, ..., n\}, \forall k \in T^i$,

(c) if $\{k_t\}$ is a sequence of elements of T^i that tends to infinity, then $\lim_{t \to \infty} \tau_j^i(k_t) = +\infty$ for every j.

For further details about asynchronous iterative algorithms the reader is referred to [5]-[8]. More specifically, convergence of asynchronous iterative algorithms has been established for many problems (see for example [5] to [16]). Particular attention must be paid to the Asynchronous Convergence Theorem of Bertsekas (see [4] and [5, p. 431]). This theorem is a powerful aid in showing convergence of asynchronous iterative algorithms and will be very useful for the design of the original termination procedure of Section three. The general conditions of the Asynchronous Convergence Theorem of Bertsekas are given as follows:

ASSUMPTION 2.1. There is a sequence of nonempty sets $\{P(k)\}$ with

$$...P(k+1) \subset P(k) \subset ... \subset R^n,$$

satisfying the following two conditions:

Box Condition: for every k, there exist sets $P_i(k), i = 1, ..., n$, such that $P(k)$ is their Cartesian product:

$$P(k) = P_1(k) \times ... \times P_n(k).$$

Synchronous Convergence Condition: $F(p) \in P(k+1)$, $\forall k$ and $p \in P(k)$. Furthermore, if $\{p(k)\}$ is a sequence such that $p(k) \in P(k)$ for every k, then every limit point of $\{p(k)\}$ is a fixed point of F.

It follows from the Asynchronous Convergence Theorem of Bertsekas that if the above conditions are satisfied and the initial solution estimate $p(0)$ belongs to the set $P(0)$, then every limit point of $\{p(k)\}$ is a fixed point of F.

2.2. Some Basic Termination Methods

An interesting approach to the termination problem has been proposed by Bertsekas and Tsitsiklis (see [5, Section 8] and [17]). The termination problem is decomposed into two parts. First, the asynchronous iterative algorithm is modified so that it converges to a fixed point sufficiently close to the solution of the problem and terminates in finite time. Then, a termination detection procedure is applied. Bertsekas and Tsitsiklis propose the following modification of the asynchronous iterative algorithm. If an update does not sufficiently alter the value of a component of the iteration vector, then it is not taken into account and communicated to other processors. Thus, termination occurs when for all processors an update causes no change in the value of the components (i.e. all local termination conditions are satisfied) and no message is in transit. Several procedures can be used for the detection of the termination. We can quote for example the procedure of Dijkstra and Scholten (see [18], and [5]) and the snapshot algorithm of Chandy and Lamport (see [19], and [5]). The former is based on message acknowledgment and generation of activity graph. The latter is based on marker message generation, with processors and links state recording at delivery of marker messages. The state recorded in a snapshot does not correspond to a true global state of the computing system at a given time. However the information contained in a snapshot is sufficient for the detection of certain properties of the global state of the system. To the best of our knowledges, the method of Bertsekas and Tsitsiklis is one of the rare methods in the literature for which there exists a rigorous proof of validity. However the method presents several drawbacks: it requires the use of a complex protocol and twice as much messages as the initial asynchronous iterative algorithm. Finally, hypotheses which are stronger than the conditions of the Asynchronous Convergence Theorem of Bertsekas and Tsitsiklis must be satisfied so that the modified asynchronous iterative algorithm converges and terminates in finite time.

Savari and Bertsekas (see [20]) have proposed another very interesting approach whereby asynchronous iterative algorithms stay essentially unchanged. Each processor sends a new update if the value of a component is changed. Another feature of this approach is the broadcast of requests in the whole network as often as the local termination condition is not satisfied. As a consequence, a processor iterates (i.e. is active) as long as its local termination condition is unsatisfied or requests are received. Termination occurs when all local termination conditions are satisfied and no request is in transit. Termination is detected using a standard protocol (see [3], [18], and [19]). Savari and Bertsekas have given a formal proof of validity of this termination method. As an advantage, this method can be applied with success to a wider class of algorithms than the method of Bertsekas and Tsitsiklis. Its main drawback is a very large number of communications. Requests must be broadcasted in the whole network whereas acknowledgments are only sent to adjacent processors in [5].

3. A NEW TERMINATION METHOD

In this section we present a new method for terminating asynchronous iterative algorithms. The method is based on a convergence concept proposed by Bertsekas (see [4] and [5, p. 431]). In particular, we make extensive use of the sequence of level sets $\{P(k)\}$ and the Asynchronous Convergence Theorem of Bertsekas presented in Section 2. The termination method requires slight modifications of the asynchronous iterative algorithms and few extra control messages.

3.1. Principle of Termination

Termination must occur when the iteration vector belongs to the level set $P(\hat{q})$, where $\hat{q}$ is a given constant and no message is in transit. We note that the termination test:

$$p(k+1) \in P(\hat{q}), \tag{3.1}$$

is particularly relevant in the case where F is a P-contraction mapping, i.e. when there exists a nonnegative matrix P with spectral radius $\rho(P) < 1$, satisfying:

$$\|F(p) - F(p')\| \leq P. \|p - p'\|, \forall p, p' \in D(F),$$

where $\|.\|$ is a vectorial norm on R^n and $D(F)$ is the domain of F. Then, the initial error satisfies:

$$\|p(0) - p^*\| \leq (I - P)^{-1} \|p(0) - F(p(0))\|,$$

where p^* denotes the solution of the problem (see [8]). Moreover the error is reduced at least by $P^{\hat{q}}$ when the termination test is satisfied.

In the sequel we will use the following notations.

The subset of T^i with elements smaller than k is denoted by $T^i(k)$. We have:

$$T^i(k) = \{r \in T^i | r < k\}.$$

N^i denotes the subset of $\{1, ..., n\}$, such that $j \in N^i$ if F_i is a function of the component p_j of vector p.

For all $i \in \{1, ..., n\}$, $\{q_i(k+1)\}$ is a subsequence of integers defined recursively by:

$$q_i(0) = 0, q_i(k+1) = \min_{j \in N^i} q_j(\tau_j^i(k)) + 1, k \in T^i. \tag{3.2}$$

We state now a preliminary result using (3.2).

LEMMA 3.1. Let Assumption 2.1 hold, $\hat{q}$ be a given constant, and consider asynchronous iterative algorithms defined by (2.1). Then, for all $i \in \{1, ..., n\}$, there exists a finite time $k \in T^i$, such that we have: $p_i(k+1) \in P_i(\hat{q})$.

Proof: If Assumption 2.1 holds then, it follows from (3.2) that $q_i(k+1)$ corresponds to the index q' of the smallest level set $P(q')$, such that $p_i(k+1) \in P(q')$. Hence, for all $i \in \{1, ..., n\}$, the sequence $\{p_i(k+1)\}$ is such that $p_i(k+1) \in P_i(q_i(k+1))$. Moreover it follows from assumptions (a)-(c) and the Asynchronous Convergence Theorem of Bertsekas that we have:

$$\lim_{k \to \infty} q_i(k+1) \to +\infty, \forall i \in \{1, ..., n\}.$$

Thus, for any given constant $\hat{q}$ and for all $i \in \{1, ..., n\}$, there exists a finite time $k \in T^i$, such that we have:

$$p_i(k+1) \in P_i(\hat{q}).$$

3.2. The Termination Method

We modify slightly the asynchronous algorithm defined by (2.1). The definition of the modified asynchronous algorithm is given as follows.

DEFINITION 3.1. Each processor implementing the modified asynchronous algorithm computes at each $k \in T^i$, $p_i(k+1)$ using (2.1) and the associated index $q_i(k+1)$ defined by (3.2). The values $p_i(k+1)$ and $q_i(k+1)$ are transmitted to adjacent processors if $\max_{r \in T^i(k)} q_i(r+1) < \hat{q}$.

We note that the computation of $q_i(k+1)$ is not time consuming. We see that the modified asynchronous algorithm requires longer messages than asynchronous iterations defined by (2.1). However, only one integer is added to each message. Moreover the last value of $p_i(k+1)$ and $q_i(k+1)$ is communicated only once.

For the termination detection we can apply a classical method like the snapshot algorithm of Chandy and Lamport (see [19]).

Finally, we note that the termination method studied in this Section can also be used for block asynchronous iterations.

The following proposition is the main result of this Section.

PROPOSITION 3.1. Let assumptions of Lemma 3.1 hold. Then, for any modified asynchronous iterative algorithm satisfying Definition 3.1, there exists a finite time $\hat{k}$ such that the termination condition holds at $\hat{k}$.

Proof: It follows from Definition 3.1 and Lemma 3.1 that for any given constant $\hat{q}$ and for all $i \in \{1, ..., n\}$, there exists a finite time $k' \in T^i$ such that we have:

$$p_i(k'+1) \in P_i(q_i(k'+1)) \subset P_i(\hat{q}),$$

where $p_i(k'+1)$ and $q_i(k'+1)$ are the last communicated values of p_i and q_i, respectively. Moreover it follows from (c) that $p_i(k'+1)$ and $q_i(k'+1)$ are not in transit after a finite time $k" > k'$.

Hence, it follows that for any given constant $\hat{q}$, there exists a finite time $\hat{k}$ such that we have:

$$p(\hat{k}) \in P(\hat{q}),$$

and no message is in transit after $\hat{k}$.

We note that the chaotic behavior of asynchronous iterations may lead to: $q_i(k'+1) \geq \hat{q}$.

The method presented in this Section can be set between the method of Bertsekas and Tsitsiklis (see [5], [17], and subsection 2.2) which alters significantly asynchronous iterations and the method of Savari and Bertsekas (see [20] and subsection 2.2) which does not modify asynchronous iterations. The main feature of the termination method presented here is that asynchronous algorithms are slightly modified. As a result, no complex procedure must cope with important modifications of the asynchronous algorithm as in [5] and [17] or a large number of communications as in [20]. Moreover the new termination procedure applies to a greater class of algorithms than the method studied in [5] and [17].

References

[1] C. Hazary and H. Zedan, A distributed algorithm for distributed termination, *Information Processing Letters*, 24 (1987), 293-297.

[2] F. Mattern, Algorithms for distributed termination detection, *Distributed Computation*, 2 (1987), 161-175.

[3] R.W. Topor, Termination detection for distributed computation, *Inform. Process. Lett.*, 18 (1984), pp. 33-36.

[4] D.P. Bertsekas, Distributed asynchronous computation of fixed points, *Mathematical Programming*, 27 (1983), 107-120.

[5] D.P. Bertsekas and J. Tsitsiklis, *Parallel and Distributed Computation, Numerical Methods*, Prentice Hall, Englewood Cliffs, New Jersey, 1989.

[6] G.M. Baudet, Asynchronous iterative methods for multiprocessors, *J. Assoc. Comput. Mach.*, 2 (1978), 226-244.

[7] D. Chazan and W. Miranker, Chaotic relaxation, *Linear Algebra Appl.*, 2 (1969), 199-222.

[8] J.C. Miellou, Algorithmes de relaxation chaotique à retards, *RAIRO*, R1 (1975), 55-82.

[9] D.P. Bertsekas and D. El Baz, Distributed asynchronous relaxation methods for convex network flow problems, *SIAM J. on Control and Optimization*, 25 (1987), 74-85.

[10] D. El Baz, M-functions, and parallel asynchronous algorithms, *SIAM Journal on Numerical Analysis*, 27 (1990), 136-140.

[11] D.J. Evans and W. Deren, An asynchronous parallel algorithm for solving a class of nonlinear simultaneous equations, *Parallel Computing*, 17 (1991), 165-180.

[12] D. El Baz, Asynchronous gradient algorithms for a class of convex separable network flow problems, to appear in *Computational Optimization and Applications.*

[13] B. Lubachevsky and D. Mitra, A chaotic asynchronous algorithm for computing the fixed point of a nonnegative matrix of unit spectral radius, *J.A.C.M.*, 33 (1986), 130-150.

[14] J.C. Miellou, Itérations chaotiques à retards, étude de la convergence dans le cas d'espaces partielle-

ment ordonnés, *C.R.A.S. Paris*, 280 (1975), 233-236.

[15] J.C. Miellou, Asynchronous iterations and order intervals, in *Parallel Algorithms and Architectures*, M. Cosnard ed., North Holland, 1986 pp. 85-96.

[16] J. C. Miellou and P. Spiteri, Un critère de convergence pour des méthodes générales de point fixe, *R.A.I.R.O. M2AN*, 19 (1985), 645-669.

[17] D.P. Bertsekas and J. Tsitsiklis, Some aspects of parallel and distributed iterative algorithms. A survey, *Automatica,* vol 27 (1991), 3-21.

[18] E.W. Dijkstra and C.S. Scholten, Termination detection for diffusing computation, *Information Processsing Letters*, 11 (1980), 1-4.

[19] K.M. Chandy and L. Lamport, Distributed snapshots: Determining global state of distributed systems, *ACM Trans. on Computer Systems,* 3 (1985), 63-75.

[20] S.A. Savari and D.P. Bertsekas, Finite termination of asynchronous iterative algorithms, *LIDS report P-2245,* M.I.T. 1994.