Appl. Math. Lett. Vol. 2, No. 2, pp. 167–170, 1989
Printed in Great Britain.

0893-9659/89 $3.00 + 0.00

A Comparison of Jacobi and Gauss–Seidel Parallel Iterations

JOHN N. TSITSIKLIS

Massachusetts Institute of Technology

Abstract. We consider an iterative algorithm in which several components are updated in parallel at each stage. We assume that the underlying iteration mapping is monotone and we show that the speed of convergence is maximized when all components are updated at each iteration.

1. INTRODUCTION.

Consider an iteration of the form $x := Ax + b$ for solving the system of linear equations $x = Ax + b$. If A is a nonnegative matrix, it is well-known that the convergence rate of the Jacobi-type iteration (all components are simultaneously updated) is no better than the convergence rate of the corresponding Gauss–Seidel iteration (components are updated one at a time). When it comes to parallel implementation, the Gauss–Seidel iteration could have certain disadvantages because variables that depend on each other can only be updated sequentially, whereas each Jacobi-type iteration takes a single step. Using the standard coloring technique [1], a Gauss–Seidel update of all components can be often performed in a reasonably small number of stages. In particular, when the matrix A is sparse, the Gauss–Seidel iteration is often amenable to massive parallelization and its improved convergence properties suggest that it might be preferable than the Jacobi iteration. Despite that, Smart and White [2] have recently shown that the parallel implementation of the Gauss–Seidel iteration cannot be faster than its Jacobi counterpart. In this note, we generalize their result by considering a) general monotone iterations and b) iterative algorithms that are intermediate between the Jacobi and Gauss–Seidel methods. Our result is very easy to derive but seems to have interesting practical implications.

2. COMPARISON RESULTS.

We consider a generic iterative algorithm of the form $x := f(x)$, where f is a function from $\Re^n$ into itself, and whose ith component is denoted by f_i. Let U be a function defined on the nonnegative integers and such that $U(t) \subset \{1, \ldots, n\}$ for each t. We interpret $U(t)$ as the set of components of x that are updated (in parallel) at stage t. We are thus concerned with the iteration

$$x_i^U(t+1) = f_i\big(x^U(t)\big), \qquad i \in U(t), \tag{1}$$

$$x_i^U(t+1) = x_i^U(t), \qquad i \notin U(t), \tag{2}$$

where $x^U(0)$ is a given initialization.

A Jacobi-type iteration corresponds to the choice $U(t) = \{1, \ldots, n\}$ for every t. We use the superscript J to indicate the sequence generated by the Jacobi-type iteration. In particular, we have $x^J(t+1) = f\big(x^J(t)\big)$, for every t. We notice that a parallel implementation of a Gauss–Seidel variant of the iteration $x := f(x)$ can be always put in the form of Eqs. (1)–(2), with a suitable choice of the sets $U(t)$.

Assumption 1: a) There exists a vector $x^* \in \Re^n$ satisfying $x^* = f(x^*)$ and such that $\lim_{t\to\infty} x^J(t)$

This research was supported by the NSF under Grant ECS–8552419 with matching funds from Bellcore Inc.

$= x^*$ for every choice of $x^J(0)$.
b) The function f is *monotone*, that is, if $x \leq y$ then $f(x) \leq f(y)$. (All inequalities between vectors are to be interpreted componentwise.)

Proposition 1: Suppose that $x(0) = x^J(0) = x^U(0)$ and that the inequality $f(x(0)) \leq x(0)$ holds. Then,

$$x^* \leq x^J(t) \leq x^U(t), \qquad \forall t \geq 0. \tag{3}$$

[A symmetrical result holds if $f(x(0)) \geq x(0)$.]

Proof: We have $x^J(1) = f(x(0)) \leq x(0) = x^J(0)$. Using the monotonicity of f, we obtain $x^J(2) = f(x^J(1)) \leq f(x^J(0)) = x^J(1)$. Proceeding similarly by induction, we obtain

$$x^J(t+1) \leq x^J(t), \qquad \forall t \geq 0. \tag{4}$$

Since $x^J(t)$ converges to x^* (by Assumption 1) we also obtain $x^* \leq x^J(t)$ for all t.

We now prove Eq. (3) by induction on t. It is certainly valid for $t = 0$. Assuming that it holds for some t, we prove it for $t+1$. We distinguish two cases:
(i) If $i \notin U(t)$, then $x_i^J(t+1) \leq x_i^J(t) \leq x_i^U(t) = x_i^U(t+1)$, where the first inequality follows from Eq. (4).
(ii) If $i \in U(t)$, we use the induction hypothesis $x^J(t) \leq x^U(t)$ and the monotonicity of f to obtain $x_i^J(t+1) = f_i(x^J(t)) \leq f_i(x^U(t)) = x_i^U(t+1)$. **Q.E.D.**

Proposition 1 shows that for any initial conditions satisfying $f(x(0)) \leq x(0)$ or $x(0) \leq f(x(0))$, the Jacobi-type iteration converges faster. The next result considers a more general class of initial conditions.

Proposition 2: Suppose that $x^J(0) = x^U(0) = x(0) > x^*$. Furthermore, suppose that there exists some $\overline{x} \geq x(0)$ such that $f(\overline{x}) \leq \overline{x}$. Then, there exists an integer K [depending on $x(0)$ and $\overline{x}$] such that $x^* \leq x^J(t+K) \leq x^U(t)$ for all t. [A symmetrical result holds if $x(0) < x^*$.]

Proof: We define $\overline{x}(t)$ by letting $\overline{x}(0) = \overline{x}$ and $\overline{x}(t+1) = f(\overline{x}(t))$. Using the monotonicity of f, the inequality $x^J(0) \leq \overline{x}$, and an easy induction, we obtain

$$x^J(t) \leq \overline{x}(t), \qquad \forall t \geq 0. \tag{5}$$

Let K be such that $\overline{x}(K) \leq x(0)$. Such a K exists because $x^* < x(0)$ and $\overline{x}(K)$ converges to x^* [Assumption 1(a)]. Let $\underline{x}(t)$ be the sequence generated according to Eqs. (1)–(2) but with $x(0)$ replaced by $\overline{x}(K)$. Since $\overline{x}(K) \leq x(0)$, the monotonicity of f implies that $\underline{x}(1) \leq x^U(1)$ and, proceeding inductively, we obtain

$$\underline{x}(t) \leq x^U(t), \qquad \forall t \geq 0. \tag{6}$$

Similarly with the proof of Eq. (4), we have $\overline{x}(t+1) \leq \overline{x}(t)$ for all t. In particular, $f(\overline{x}(K)) = \overline{x}(K+1) \leq \overline{x}(K)$. Notice that the sequence $\{\overline{x}(t+K) \mid t = 0, 1, \ldots\}$ is generated by the Jacobi iteration starting (at $t = 0$) at the vector $\overline{x}(K)$. The sequence $\underline{x}(t)$ is initialized at the same vector but components are updated as determined by U. Since $f(\overline{x}(K)) \leq \overline{x}(K)$, Prop. 1 applies and shows that $\overline{x}(t+K) \leq \underline{x}(t)$. Combining this inequality with Eqs. (5) and (6), we obtain

$$x^J(t+K) \leq \overline{x}(t+K) \leq \underline{x}(t) \leq x^U(t), \qquad \forall t \geq 0.$$

Q.E.D.

Proposition 2 shows that if $x(0)$ satisfies $x(0) > x^*$ or $x(0) < x^*$ then $x^J(t)$ can "lag behind" $x^U(t)$ by at most a constant number K of steps. An easy corollary is that the

asymptotic convergence rate of $x^J(t)$ is no worse than that of $x^U(t)$, if $x(0)$ is as above. For example, if we assume that $x^J(t)$ converges at the rate of geometric progression and, in particular, that

$$\lim_{t\to\infty} \left(\|x^J(t) - x^*\|_\infty\right)^{1/t} = \rho,$$

it is easily shown that

$$\liminf_{t\to\infty} \left(\|x^U(t) - x^*\|_\infty\right)^{1/t} \geq \rho.$$

3. APPLICATIONS.

Suppose that f is of the form $f(x) = Ax + b$ where A is an $n \times n$ irreducible nonnegative matrix and $b \in \Re^n$, and let x^* satisfy $x^* = Ax^* + b$. Under the assumption that the Jacobi–type algorithm converges, the spectral radius $\rho(A)$ of A is less than 1 and (by the Perron–Frobenius theorem) there exists a positive vector w such that $Aw = \rho(A)w < w$. It is seen that Assumption 1 is satisfied and, if the initialization $x(0) = w + x^*$ is used, we have $f(x(0)) = A(x^* + w) + b = x^* + Aw = x^* + \rho(A)w < x^* + w = x(0)$. Therefore, Prop. 1 applies and yields $x^U(t) - x^* \geq x^J(t) - x^* = \rho(A)^t w$. In particular, the convergence rate of $x^U(t)$ can be no better than the convergence rate $\rho(A)$ of the Jacobi–type iteration and we have recovered the result of [2].

There are several situations in which the iteration mapping f is nonlinear and satisfies our assumptions, e.g. in dynamic programming or nonlinear optimization [1]. One example is a variant of the "nonlinear Jacobi" algorithm (incorporating an underrelaxation parameter) for the solution of the dual of a strictly convex network flow problem (see [1] for a description of the algorithm and its properties). A Gauss–Seidel variant of this algorithm has been studied in [3] and has been tested in a parallel environment [4]. Our result shows that the Jacobi variant is actually preferable, and this is consistent with what was observed in the experiments reported in [4].

Another interesting example is the Bellman–Ford algorithm for finding shortest paths in networks. Here, we are given a directed graph $G = (V, E)$, with node set $V = \{1, \ldots, n\}$ and edge set E. Also, for each $(i, j) \in E$, we are given a scalar a_{ij} representing the length of arc (i, j). Let 1 be a destination node, and we are interested in finding the length of a shortest path from any node $i \neq 1$ to node 1. The Bellman–Ford algorithm finds the shortest distances (assuming that they are finite) by means of the iteration

$$x_i := \min_{\{j \mid (i,j)\in E\}} \{a_{ij} + x_j\}, \qquad i \neq 1, \tag{7}$$

while x_1 is fixed to 0. The algorithm is guaranteed to converge starting from any initial conditions, the iteration mapping is clearly monotone, and Assumption 1 holds. The standard initialization of the algorithm is to let $x(0) = \overline{x}$, where $\overline{x}_i = \infty$ for every $i \neq 1$, and $\overline{x}_1 = 0$. The inequality $f(\overline{x}) \leq \overline{x}$ is trivially true and Prop. 1 applies. (There is a minor issue because the vector $\overline{x}$ does not belong to $\Re^n$. However, the proof of Prop. 1 goes through verbatim for this case.) We conclude that the parallel Jacobi version of iteration (7) is no slower than any parallel Gauss–Seidel variant of that iteration. This is in sharp contrast to what happens in serial computing environments in which Gauss–Seidel variants are known to substantially outperform the Jacobi iteration.

All of our discussion has been based on the implicit assumption that there are n processors available so that an iteration of the Jacobi algorithm can be performed in parallel, in a single step. It should be emphasized that our results are not relevant to the case where fewer than n processors are available; in particular, when the number of processors is sufficiently small the Gauss–Seidel variant can be shown to be preferable. Similarly, no general statement can be made for the case of non–monotone iterations. For instance, there are numerous algorithms whose Jacobi variant fails to converge but their Gauss–Seidel variant converges and is therefore preferable.

REFERENCES

1. D.P. Bertsekas, J. N. Tsitsiklis, "Parallel and Distributed Computation: Numerical Methods," Prentice Hall, Englewood Cliffs, NJ, 1989.
2. D. Smart, J. White, *Reducing the Parallel Solution Time of Sparse Circuit Matrices Using Reordered Gaussian Elimination and Relaxation*, Proceedings of the 1988 ISCAS, Espoo, Finland.
3. S.A. Zenios, J.M. Mulvey, *A Distributed Algorithm for Convex Network Optimization Problems*, Parallel Computing 6 (1988), 45–56.
4. S.A. Zenios, R.A. Lasken, *Nonlinear Network Optimization on a Massively Parallel Connection Machine*, Annals of Operations Research 14 (1988).

Massachusetts Institute of Technology
Cambridge, Massachusetts 02139