Asynchronous iterations for the solution of Markov systems

Mohamed Jarraya, Didier El Baz *

LAAS du CNRS, 7, avenue du Colonel Roche, 31077 Toulouse Cedex 4, France

1 Introduction

Parallel asynchronous iterative algorithms for the solution of Markov systems have been proposed in ([2],[3] and[4]). We are interested here in the study of asynchronous iterative algorithms for the solution of a block tridiagonal Markov system that occurs in circuit-switched network modeling i.e. parallel algorithms whereby synchronisation among processors of a parallel computer is not performed at the end of each iteration.

2 Sequential iterative methods

We consider the following Markov systems :

$$QP = 0 \tag{1}$$

where $P = [P_0, ..., P_{n-1}]$ is a stationnary probability vector of dimension n, with $\sum_{k=0}^{n-1} P_k = 1$ and Q is a block tridiagonal irreducible, aperiodic, Z-matrix such that $\sum_{k=0}^{n-1} q_{ki} = 0, \forall i$. Under these assumptions, there is a single vector P^* solution of (1) such that $\sum_{i=0}^{n-1} P_i^* = 1$ (see[3]). The matrix Q is decomposed into n blocks. In the following, z will represent the number of states of the system (i.e. the dimension of the vector P), z_k the number of states of the block k, $0 \leq k \leq n-1$ with $\sum_{k=0}^{n-1} z_k = z$ and P_k the probability vector associated with the block k, $P_k^T = [\pi_{k,1}, ..., \pi_{k,z_k}]$.
We can use an iterative method such as the over relaxation method to solve the linear system (1). A different approach proposed by Takahashi (see [6]) called aggregation-disagregation computes the conditional probabilities of the blocks and generates another block tridiagonal system. Each method is based on the Gauss-Seidel iterative scheme.

* The authors wish to thank IDRIS-CNRS for his support.

3 Parallel iterative methods for the solution of Markov system

Let us consider the following splitting of Q : $Q = D - L - L' - U - U'$

$$Q = \begin{pmatrix} (D_0 - L_0 - U_0) & -U'_0 & & 0 \\ -L'_1 & (D_1 - L_1 - U_1) & & \\ & & \ddots & -U'_{\alpha-2} \\ 0 & & -L'_{\alpha-1} & (D_{\alpha-1} - L_{\alpha-1} - U_{\alpha-1}) \end{pmatrix}$$

$$D_i = \begin{pmatrix} A_{N_{(i)}} & 0 & \dots & 0 \\ 0 & A_{N_{(i)}+1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & A_{N_{(i+1)}-1} \end{pmatrix} \quad L_i = \begin{pmatrix} 0 & 0 & \dots & 0 \\ -B_{N_{(i)}+1} & \ddots & & \vdots \\ & \ddots & \ddots & 0 \\ 0 & & -B_{N_{(i+1)}-2} & 0 \end{pmatrix}$$

$$L'_i = \begin{pmatrix} 0 & -B_{N_{(i)}} \\ \vdots \ddots & \\ 0 \dots & 0 \end{pmatrix}, U_i = \begin{pmatrix} 0 & -C_{N_{(i)}} & 0 \\ \vdots & \ddots & \ddots \\ & & -C_{N_{(i+1)}-3} \\ 0 & \dots & 0 \end{pmatrix}, U'_i = \begin{pmatrix} 0 & \dots 0 \\ & \vdots \\ -C_{N_{(i+1)}-1} & 0 \end{pmatrix}$$

where $A_i = Q_{i,i}$, $B_i = Q_{i,i-1}$, $C_i = Q_{i,i+1}$, $\alpha < n$ is the number of processors, we have $\sum_{j=0}^{\alpha-1} n_j = n$, where n_j is the dimension of the $j-th$ subsytem assigned to processor j and $N_{(i)} = \sum_{j=0}^{i-1} n_j$.
The classical Gauss-Seidel matrix is given by $G = (D - L - L')^{-1}(U + U')$.
The parallel Gauss-Seidel like matrix is : $Gp = (D - L - U')^{-1}(U + L')$.
P. AMODIO and F. MAZZIA have showed in [1] that the matrices G_p and G have the same spectral radius. The parallel Gauss-Seidel like algorithm is as follows:
we suppose that $B_0 \equiv C_{n-1} \equiv 0$

if $i \neq \alpha - 1$, then send $P^{(m-1)}_{N_{(i+1)}-1}$ to the processor $i + 1$
if $i \neq 0$, then receive $P^{(m-1)}_{N_{(i)}-1}$ from processor $i - 1$
$P^{(m)}_{N_{(i)}} = A^{-1}_{N_{(i)}}(-B_{N_{(i)}} P^{(m-1)}_{N_{(i)}-1} - C_{N_{(i)}+1} P^{(m-1)}_{N_{(i)}+1})$
for $j = N_{(i)} + 1,\ N_{(i+1)} - 2$
$P^{(m)}_j = A_j^{-1}(-B_j P^{(m)}_{j-1} - C_j P^{(m-1)}_{j+1})$
end for
if $i \neq 0$, then send $P^{(m)}_{N_{(i)}}$ to the processor $i - 1$
if $i \neq \alpha - 1$, then receive $P^{(m)}_{N_{(i+1)}}$ from processor $i + 1$
$P^{(m)}_{N_{(i+1)}-1} = A^{-1}_{N_{(i+1)}-1}(-B_{N_{(i+1)}-1} P^{(m)}_{N_{(i+1)}-2} - C_{N_{(i+1)}-1} P^{(m)}_{N_{(i+1)}})$

Note that the parallel Gauss-Seidel like algorithm is used both for over relaxation and aggregation-disaggregation method and it can be used for all methods like $x^{(m)} = f(x^{(m)}, x^{(m-1)})$, where f is a linear function.

We present now a model of asynchronous iterative algorithms derived from the model of Lubachevsky and Mitra [4] and similar to the model of Miellou [5].

Definition 1. *Let $\mathcal{N}$ be the set of natural numbers, n, $\alpha \in \mathcal{N}$, with $\alpha \leq n$ and the decomposition of R^n into $\Pi_{i=0}^{\alpha-1} R^{n_i}$ with $\sum_{i=0}^{\alpha-1} n_i = n$. An asynchronous iteration associated with the fixed point equation $MP = P$ and the initial point $P^{(0)} \in \Pi_{i=0}^{\alpha-1} R^{n_i}$ is a sequence $P^{(m)}$; $m = 0, 1...$ of vectors of $\Pi_{i=0}^{\alpha-1} R^{n_i}$ defined recursively by :*

$$\begin{cases} P_i^{(m+1)} = \sum_{j=0}^{\alpha-1} M_{ij} P_j^{(m-d(m,i,j))} & \text{if } i \in S(m), \\ P_i^{(m+1)} = P_i^{(m)} & \text{if } i \notin S(m), \end{cases}$$

where the delay terms $d(m,i,j)$ depends on the computation times, communications , memory access and software delays, $\mathcal{S} = \{S(m) | m = 0, 1, ...\}$ is a sequence of nonempty subsets of $\{0, 1, ...\alpha - 1\}$.

Assumptions :
A1 : There exists a finite d which uniformly bounds the delay terms, i.e.
$0 \leq d(m,i,j) \leq d < +\infty$; $m \geq 0$, $0 \leq i \leq \alpha - 1$, $0 \leq j \leq \alpha - 1$.
A2 : There exists a finite integer s, such that every coordinate is updated at least once in every s consecutive updates.

4 Implementation on the CRAY T3E

The various synchronous and asynchronous iterations were implemented on the supercomputer CRAY T3E of IDRIS-CNRS. In the synchronous case we have used the functions MPI_Isend() and MPI_Recv() of the MPI library in order to exchange data betwen processors. Synchronization is guaranteed by the blocking function MPI_Recv() which can end only when one message is actually received. In the asynchronous case, each processor computes at its own pace without waiting for the value of data coming from other processors. Except for communication primitives, the basic procedure is very similar to the parallel Gauss-Seidel like algorithm. Data exchange between processors is done by diffusing the new updates to the processors which must use them in their computation. This policy uses the put() function of the SHMEM library of CRAY.

5 Numerical results

We have chosen a matrix Q with dimension 9900x9900 and having n=36 diagonal blocks of variable size. The precision of termination is $\epsilon = 0.0001$ in the sequential and parallel asynchronous iterative algorithms. Table 1 shows, for the various sequential, parallel synchronous and asynchronous iterative methods, the number of iterations, the execution time and the efficency. In the sequential case, the aggregation-disaggregation method is faster than the over relaxation method applied to system (1). The parameter of the over relaxation method is $\omega = 1.6$ and for the aggregation-disaggregation method is $\omega = 1.5$. The comparison of synchronous and asynchronous parallel iterations, shows that asynchronous iterations are faster than synchronous iterations. This proves that load ambalancing caused by the variable size of the blocks adds a waiting time to synchronize processors.

	Synchronous			Asynchronous		
Number of processors	Number of iterations	Execution time	Efficiency	Number of iterations	Execution time	Efficiency
	Over relaxation method					
1	59	24.047	1	59	24.047	1
2	59	12.986	0.925	67	12.866	0.934
4	60	8.345	0.720	119	8.151	0.737
6	58	6.715	0.596	151	6.740	0.593
9	57	4.590	0.582	217	4.291	0.622
	Aggregation-disaggregation method					
1	51	20.960	1	51	20.960	1
2	52	11.536	0.908	58	11.208	0.935
4	52	7.282	0.719	103	7.116	0.736
6	52	6.058	0.576	131	5.916	0.590
9	54	4.378	0.531	183	3.762	0.619

Table 1. Iterative methods

References

1. Amodio, P., Mazzia, F.: A parallel Gauss-Seidel method for block tridiagonal linear systems. Siam J. Sci. Comput, 6:1451-1461, Novembre 1995.
2. Bernussou, J., Le Gall, F.: About some iterative synchronous and asynchronous methods for Markov chain distribution computation. IFAC' 87 - 10^{th} world congress - Munich - 27, 31 Juillet 1987.
3. El Baz, D.: Parallel Iterative Algorithms for the Solution of Markov Systems. Proceedings of the 33rd IEEE Conference on Decision and Control, Orlando USA, 2524-2527, 1994.
4. Lubachevsky, B., Mitra, D.: A chaotic asynchronous algorithm for computing the fixed point of a non negative matrix of unit spectral radius. J.A.C.M., 33, 1:130-150, 1986.
5. Miellou, J.C.: Algorithmes de relaxation chaotique à retard. R.A.I.R.O., 1:55-82, 1975.
6. Takahashi, Y.: A lumping method for numerical calculations of stationnary distributions of Markov chains. Research report B.18 -17 Department of Information Sciences, Tokyo Institute of Technology, 1975.