Bernadette L. Kirk [1] and Yousry Y. Azmy [2]

An Optimized Algorithm for Solving the Nodal Diffusion Method on Shared Memory Multiprocessors*

Bernadette L. Kirk [1] and Yousry Y. Azmy [2]

Oak Ridge National Laboratory
P.O. Box 2008
Oak Ridge, TN 37831-6362

Nodal methods play a special role in reactor physics calculations. In recent papers the high computational efficiency of nodal methods has been established [1] and the development of more efficient algorithms tailored to the advanced architectures of modern day computers proposed [2-6]. The rapidly changing architectures of today's computer influence the way codes have to be programmed so that reasonable speed up and efficiency are attained. We have applied these concepts in solving the one-group neutron diffusion equation in two-dimensional geometry on parallel computers like the Intel iPSC/2 hypercube and the Sequent Balance 8000. The efficiency of the hypercube for the neutron diffusion equation is highly determined by the message passing scheme; on the other hand, on a shared memory processor like the Sequent, it is dependent on the manipulation of variables in shared memory. In this paper, we present a scheme on shared memory processors which produces very high computing efficiencies in agreement with Amdahl's law.

We start with the one-group neutron diffusion equation in two-dimensional geometry given by

$$D\left[\frac{\partial^2\phi}{\partial x^2}+\frac{\partial^2\phi}{\partial y^2}\right]-\sigma\phi=-S\ , \tag{1}$$

where D is the diffusion coefficient, σ the removal cross section, S the external source and ϕ the neutron flux.

By dividing the problem geometry into a set of closed rectangular regions of the form $[-a_m,+a_m]\times[-b_m,+b_m]$, $m=1,...,M$, integrating over the volume of node m, averaging in the x- and y-directions, and proper substitution of resulting variables, a five-point scheme for the neutron balance equation is obtained

$$\frac{D_m}{2}\left\{\left(\frac{\gamma_m^2P_m^x}{1-P_m^x}\right)\left(\bar{\phi}^y_{+m}-2\bar{\bar{\phi}}_m+\bar{\phi}^y_{-m}\right)+\left(\frac{\gamma_m^2P_m^y}{1-P_m^y}\right)\left(\bar{\phi}^x_{+m}-2\bar{\bar{\phi}}_m+\bar{\phi}^x_{-m}\right)\right\}-\gamma_m\bar{\bar{\phi}}_m=-\bar{\bar{S}}_m\ , \tag{2}$$

where

$$P_m^y\equiv\frac{\tanh(\gamma_mb_m)}{\gamma_mb_m},\quad P_m^y\equiv\frac{\tanh(\gamma_ma_m)}{\gamma_ma_m},\quad \gamma_m=\sqrt{\sigma_m/D_m}\ ,$$

$\bar{\bar{\phi}}_m$'s are the node-averaged fluxes, $\bar{\phi}^y_{-m}$, $\bar{\phi}^y_{+m}$, ϕ^x_{-m}, ϕ^x_{+m} are the surface-averaged fluxes.

To obtain the current continuity conditions, integration is performed over the closed interval $[-b_m,+b_m]$, across y-constant surfaces for the y-current equations,

$$\phi^x_{+m}\left[D_m\left(\frac{1+\omega^y_m}{2b_mP^y_m}\right)\quad D_n\left(\frac{1+\omega^y_n}{2b_nP^y_n}\right)\right]-\bar{\phi}^x_{-m}D_m\left(\frac{1-\omega^y_m}{2b_mP^y_m}\right)-\bar{\phi}^x_{+n}D_n\left(\frac{1-\omega^y_n}{2b_nP^y_n}\right) \tag{3}$$

$$-\bar{\bar{\phi}}_m\left(\frac{D_m\omega^y_m}{b_mP^y_m}\right)-\bar{\bar{\phi}}_n\left(\frac{D_n\omega^y_n}{b_nP^y_n}\right)=0\ ,$$

where node n is vertically adjacent to node m, $\phi^x_{+m}=\bar{\phi}^x_{-m}$.

Similarly, if integration is done over $[-a_m,+a_m]$ across x-constant surfaces, the x-current equations become:

$$\phi^y_{+m}\left[D_m\left(\frac{1+\omega^x_m}{2a_mP^x_m}\right)+D_l\left(\frac{1+\omega^x_l}{2a_lP^x_l}\right)\right]-\phi^y_{-m}D_m\left(\frac{1-\omega^x_m}{2a_mP^x_m}\right)-\phi^y_{+l}D_l\left(\frac{1-\omega^x_l}{2a_lP^x_l}\right) \tag{4}$$

$$-\bar{\bar{\phi}}_m\left(\frac{D_m\omega^x_m}{a_mP^x_m}\right)-\bar{\bar{\phi}}_l\left(\frac{D_l\omega^x_l}{a_lP^x_l}\right)=0\ ,$$

where node l is horizontally adjacent to node m in the positive x-direction.

We describe briefly the numerical steps taken in solving Eqs. (2-4). We start with initial estimates of the node-averaged fluxes $\bar{\bar{\phi}}_m$'s. Equations (3-4) then become systems independent of each other; that is Eq.(3) only involves the unknown $\bar{\phi}^x_{+m}$, $\bar{\phi}^x_{-m}$, and $\bar{\phi}^x_{+n}$ and Eq.(4) the unknowns ϕ^y_{+m}, $\bar{\phi}^y_{-m}$, ϕ^y_{+l}. These two systems can be solved in parallel. This comprises one iteration. In the next iteration, Eq.(2) is used to update the node-averaged fluxes. These newly updated values are substituted in Eqs. (3-4) again and the parallel systems are solved. Convergence of this method is determined by examining the relative difference between the previous and the new iterates of the $\bar{\bar{\phi}}_m$'s. Successive overrelaxation is also applied in between iterations to the $\bar{\bar{\phi}}_m$'s for faster convergence. This algorithm is very well suited to computers with parallel architectures. In shared memory systems, cautious programming is needed in order that the processors do not overwrite memory locations which are vital in each iteration. This is done through locking mechanisms.

Following is a description of the implementation of the iterative method on shared memory processors. We store the old and new iterates of the node-averaged fluxes, the current iterates of the surface fluxes, the mesh size and other physical parameters needed into common memory. For an $n\times n$ mesh, there will be $2n$ independent processes. The program starts running initially on one processor, which becomes the parent process. The parent process then 'forks' and creates subprocesses. Each processor is assigned a subprocess which is the solution of the tridiagonal current continuity equations. In order for the processors to start at the same time, synchronization calls are performed before and after the solution of the current equations. Upon exiting from this step, the processors have solved the surface-averaged fluxes for given initial values of the node-averaged fluxes. Then they update specifically assigned values of the node-averaged fluxes with the

* Research sponsored by the Office of Nuclear Energy, U.S. Department of Energy, under contract DE-AC05-84OR21400 with Martin Marietta Energy Systems, Inc.

[1] Engineering Physics and Mathematics Division (EPMD), Radiation Shielding Information Center.

[2] EPMD, Advanced Systems.

DISCLAIMER

use of successive overrelaxation. Each processor J, then computes the relative difference between the old and new $\phi_{m,n}$'s assigned to it and stores the maximum value into DIFMX(J). A call to synchronize all processes is done at this point. The check for convergence is assigned to processor 0 and is done by comparing all values of DIFMX(J), J=1,...,NP (where NP is the number of processes) to a given tolerance criterion (this is the only portion of the code which is not parallelizable). When convergence is attained, all the processes are released.

The above approach to solving the nodal diffusion equations is totally lock-free. The avoidance of locks is achieved by storing the surface fluxes into common memory. The extra expense in memory is minimal.

We applied the algorithm to a test problem and the measured speedup and efficiency are presented in Fig. 1. On a 32 × 32 mesh, we achieved efficiencies of 99.2 for two processors, 99.1 for four processors and 98.6 for eight processors on a Sequent Balance 8000 with 16 megabytes of memory. The important question of load balancing via dynamic scheduling will be addressed in the full paper.

REFERENCES

1. R. D. Lawrence, "Progress In Nodal Methods for the Solution of the Neutron Diffusion and Transport Equations." *Progress In Nuclear Energy*, 17:3, 271, 1986.
2. B. L. Kirk and Y. Y. Azmy, "Hypercube Applications of the $x-y$ Geometry Nodal Method for the Neutron Diffusion Equation," *Trans. Am. Nucl. Soc.*, (1988).
3. B. L. Kirk and Y. Y Azmy, "A Parallel Approach to the Nodal Method Solution of the Two-Dimensional Neutron Diffusion Equation," *Proceedings of ANS Topical Meeting on Advances in Nuclear Engineering Computation and Radiation Shielding, Santa Fe, New Mexico, April 9-13, 1989* Vol.I.
4. Sung-Dyun Zee and Paul J. Turinsky, "Vectorized and Multitasked Solutions of the Two-Group Neutron Diffusion Equations," pp 83-94 in *Proc. of ANS Internatl. Top. Advances in Reactor Physics, Mathematics and Computation, Paris, April 27-30, 1987.*
5. H. L. Rajic and A. M. Ongonag, "A Vectorized-Concurrent Nodal Neutron Diffusion Method," p 163 in *Proc. 1988 International Reactor Physics Conference, Jackson Hole, Wyoming, Sept. 18-22, 1988* Vol. IV, American Nuclear Society, LaGrange Park, Illinois, 1988.
6. B. L. Kirk and Y. Y. Azmy, "An Iterative Algorithm for Solving the Multidimensional Neutron Diffusion Nodal Method on Parallel Computers," submitted to the *Journal of Computational Physics*, 1989.

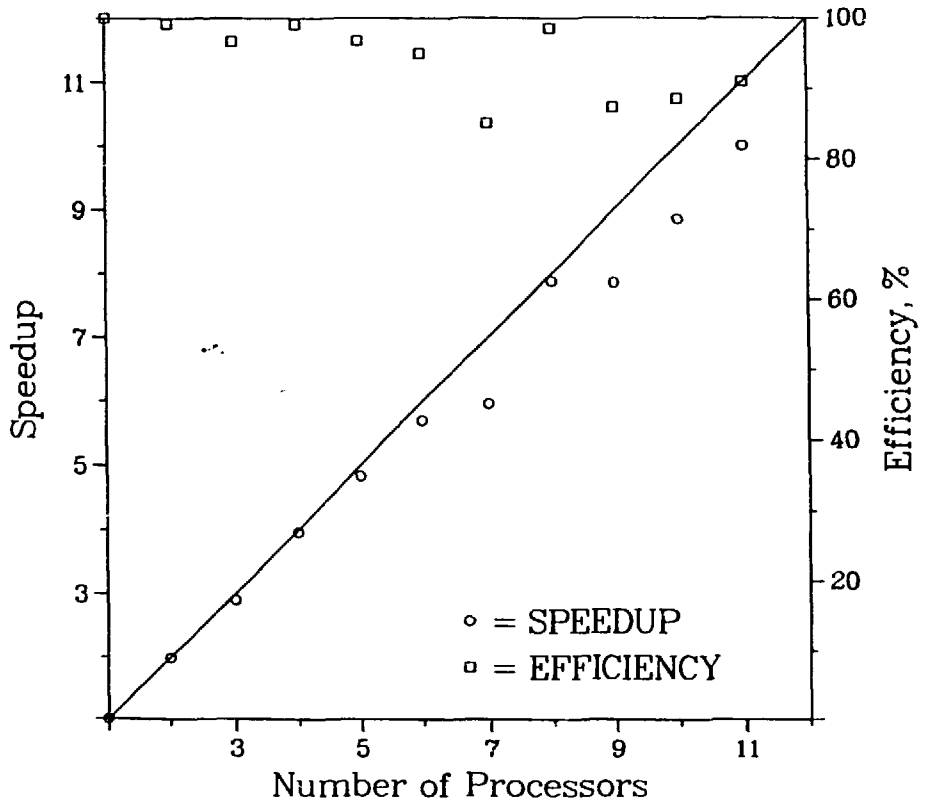

Fig. 1. The speedup and efficiency of the lock-free parallel algorithm for the 32 × 32 mesh test problem as a function of the number of participating processors on the Sequent Balance 8000.