Recent Developments in Parallel Method of Lines
Solutions of Partial Differential Equations

S. C. Chu, Radford University
W. F. Lawkins, Oak Ridge National Laboratory
E. Ng, Oak Ridge National Laboratory
S. Thompson, Radford University

Abstract

We present a summary of recent and ongoing attempts to exploit parallelism in the numerical solution of method of lines (MOL) problems in partial differential equations (PDEs). MOL is an established technique for solving PDEs [18]. The success of MOL depends in part on the choice of the spatial differentiation scheme(s) used to reduce PDEs to systems of ordinary differential equations (ODEs) and in part on the use of high quality ODE solvers to integrate the resulting systems. There are several such ODE solvers available. Some of the best include DDRIV3 [12], LSODE [10], and VODE [2]. The primary expense associated with the use of ODE solvers is the formation of the Jacobian matrix and the linear algebra required to solve the corrector equations. For dense systems, it is not uncommon for these two stages in the solution to account for virtually all of the execution time required to solve a problem in this fashion [16]. Even when problem structure (e.g., sparsity) is exploited, these operations often account for most of the execution time required.

Previous studies have considered the parallelization of the Jacobian formation and linear algebra calculations. Parallel formation of the Jacobian using finite differences is an easy matter since the finite differences are independent of one another. Furthermore, while formation of the Jacobian is an expensive operation, it is the linear algebra which dominates the time required. (See [16] for comparative results for a typical problem.) For these reasons, several previous and ongoing studies have included both but with the primary emphasis being on the latter. (In fact, an objective of recent studies has been to eliminate direct formation or use of the Jacobian [3].)

It was demonstrated in [13] that it is possible when using DDRIV3 for dense systems to achieve near asymptotic speedups for shared memory parallel machines by forming the Jacobian matrix in parallel and by replacing the usual serial direct Gaussian elimination techniques by parallel direct variants. (Similar results for LSODE may be found in [16].) It was demonstrated in [14] that for sparse systems impressive speedups are possible using parallel direct sparse Gaussian elimination techniques. Iterative serial sparse solvers from [17] have been used in conjunction with the code used in [14].

Different issues are involved with parallel MOL solutions on distributed memory computers. It is generally not possible to achieve speedups comparable to those for shared memory machines due to the required interprocess communication. However, significant speedups are possible. The use of DDRIV3 in conjunction with parallel direct Gaussian elimination methods for distributed memory machines was considered in [15] for dense systems.

Direct methods are attractive since they require a finite number of operations. However, dense methods are not feasible for extremely large problems. For sparse Jacobians, direct methods suffer fill and thus may require too much memory. Although permuting the Jacobian to preserve sparsity and numerical stability is still an active area of research, present investigations involve the use of iterative methods. While an iterative method may require an unpredictable number of operations due to its convergence behavior, use of such methods is an attractive alternative since iterative methods do not require a full matrix factorization. If an iterative method is used, the crucial sub-steps are matrix-vector multiplications and inner-product computations.
Very impressive results have been obtained by incorporating iterative linear algebra techniques in two of the best available serial ODE solvers [3,5]. The resulting “Krylov” ODE solvers, LSODPK and VODPK, have been used to solve problems successfully which are beyond the capacity of solvers based on direct Gaussian elimination. The heuristics used in these solvers represent effective guides for incorporation of parallel variants of the iterative methods used in the solvers. In fact, the developers of LSODPK and VODPK presently are considering extensions of the solvers to distributed memory machines. Other studies which consider the use of serial iterative methods and which serve as guides for the incorporation of parallel iterative methods in this context include [4,6,8,11].

The DDRIV3 solver was chosen for the studies reported in [13–15] (although the techniques are applicable to other solvers). One of the reasons for this choice is the solver’s capability of allowing the user to supply and completely control the Jacobian formation and linear algebra via the “USERS” option. (Similar capabilities are present in LSODPK and VODPK.) The use of parallel methods may then be investigated without altering the basic strategies used in the ODE solver (although the use of iterative methods may suggest ways in which an ODE solver’s internal adaptive strategies may be improved).

Presently, the use of DDRIV3 in conjunction with various iterative methods is being considered. For example, the serial iterative solvers (and parallel variants) from [1] are being considered. In addition, parallel implementations of similar iterative methods from [7] are being considered. (Preliminary results obtained using the PIM package of solvers in [7] are particularly encouraging.) One of the objectives of this approach is to couple the use of available high quality ODE solvers and parallel iterative methods since significant advances are anticipated for the latter in the near future.

In addition, the use of parallel preconditioners [1,7] is being investigated. Preconditioning is an important problem since it is often hard to get good performance out of an iterative linear equation solver without preconditioning to improve convergence properties of iterative methods [5]. The development of good parallel preconditioners for MOL appears to be one of the hardest problems to overcome in this context since problem dependencies associated with developing effective preconditioners make it difficult to design robust methods. Preconditioning also represents one way in which direct methods (particularly sparse ones) may be used in conjunction with iterative methods (for example, if incomplete LU factorizations are used as preconditioners).

A PVM [9] based version of the parallel code used in [15] has been developed and is being used in the present investigations. The PVM based version allows various approaches to be investigated on networked serial machines and later ported to distributed memory machines such as the Paragon computer. Reference [4] contains a description of the use of PVM in connection with the VODE solver and illustrates the attractiveness of this approach.

We will present some preliminary results. A great deal of work remains before it will be possible to determine the extent to which parallel methods may be exploited for method of lines problems. In fact, the most effective strategy may very well be to couple the use of iterative solvers and direct solvers, both dense and sparse, for use on sub-problems (associated with preconditioning, for example). However, the preliminary results suggest that substantial speedups are possible in this context by using parallel iterative methods.

References
