Crashworthiness Simulations by using PAM-CRASH on the IBM 9076 SP1

M. Briscolini, *J. Clinckemaillie and S. Vlachoutsis †

First European PVM Users' Group Meeting,
University of Rome, Italy, October 9-11, 1994.

The migration of full industrial applications to massively parallel processing (MPP) has recently undertaken a relevant importance in order to exploit large computational capabilities as those provided by highly scalable distributed memory architectures. This new class of parallel architectures combines high flexibility with an attractive price-performance, allowing to perform high intensive numerical simulations for several engineering applications.

Contextually, industrial crashworthiness simulations may achieve a large improvement exploiting MPP in order to reduce the throughput for the analysis of car prototypes under strong deformations and using a finer discretization of any component of car (carriage, block-engine, shock-absorber, wheel-block, air-bag, etc.).

PAM-CRASH, an explicit time-marching finite element based simulation program, is commonly used by several automotive industries for performing crashworthiness simulations. The parallel porting activity started about two years ago and was in large part supported under grant of the E.E.C. in the ESPRIT III Project CAMAS (Computer Aided Migration of Applications System). Specifically, the first part of the porting focused the parallelization of the computational kernel for the shell element force evaluations plus the time-marching scheme [3]. This is one of the most CPU time consuming kernel for crashworthiness simulations being that large part of finite elements, employed for the discretization, are shell type. Typically, in a realistic simulation, about 80% of CPU time is spent on shell elements while the remaining part accounts forces in other elements (beams, solid and trusses), and self-contact [1]. It clearly appears that an efficaciously parallelization of such part of code substantially needs a large restructuring of data organization with the aim to optimize data distribution among concurrent processors and to explicit communications. The former

*IBM ECSEC, European Center for Scientific and Engineering Computing, viale Oceano Pacifico 171, 00144, Roma, Italy

†ESI SA, Engineering System International SA, 20 Rue Saarineen, Silic 270, 94578 Rungis-Cedex, France

requires a partitioning of computational mesh in different subdomains which will be allocated on the different concurrent processors. The latter is completed by avoiding as much as possible communication delays due to global synchronizations, and introducing suited massage-passing constructs to exchange data among concurrent processors.

In order to exploit large scalability of any parallel code, it is recommended to obtain an optimal partitioning of work-load among processors which has to be provided by using an efficient Domain Decomposition Tool (hereinafter DDT) [3]. DDT will tend to enforce the same number of mesh points/elements on each subdomains minimizing the number of interface parameters between adjacent partitions In the present experience, several mathematical methods was investigated and some preliminary tests on realistic geometries have unequivocally shown that the method based on Recursive Spectral Bisection [6] is one of the most efficient. In particular, i) RSB is moderately CPU-time consuming on performing the partition, even for a very large mesh; ii) it can be automatically used without requiring any operator intervention; iii) it allows to satisfy the optimal constrains for data distribution until to decades of subdomains.

The parallelization strategy of shell force evaluation kernel has been described in detail in references [3, 4, 5] and in the following we only want to summarize the main features.

The integration method is performed in different steps. At end of an integration cycle, displacements at nodal points are known. Then, forces on elements defined by displacement of related nodal points may be calculated by using one-point Gaussian-Lobatto quadrature scheme. Nodal forces are estimated in order to compute accelerations from the equation of the dynamic equilibrium and the central difference scheme allows the calculation of displacements for a new time step [3].

Parallel implementation of shell force evaluations occurred in the following method. The domain decomposition partitions mesh along element boundaries so that the nodal points along those interfaces are doubly defined. A complete time integration needs a uni-directional communication of forces and velocities between adjacent partitions while displacement computations are duplicated at the boundary. In such a way, the force evaluation kernel achieves about 100% of parallel efficiency. At the end, after having obtained new displacements, the global stability time-step, based on the Courant criteria, is estimated by minimizing local optimal time-step among all partitions. This requires a global communication among all processes in order to compute the minimal time-step and to broadcast such information to all concurrent processes.

The first parallel implementation of shell force evaluation kernel it was based on the portable message passing interface PARMACS (release 5.1) [2]. This parallel interface is particularly useful for architectures where the concept of a hierarchical tree processor organization has some implications. At the contrary, in a parallel system as a cluster of workstations, the PVM message passing interface appears more adapted to an efficient parallel programming. Then, the code has been easily converted replacing PARMACS paradigms with the PVM ones.

The PVM version has been tested on a cluster of IBM RS/6000 and on the IBM scalable POWERparallel SP1. Tests refer to a simple geometry, a rectangular strip, where $r \times s$ are the number of elements whose interfaces are of length r. Table 1 reports speedups for a cluster of four IBM RS/6000 mod.560 interconnect by SOCC, a prototype of switch based on optical fiber, while table 2 reports data on the 16-nodes SP1, equipped by the *high performing switch* (HPS) [7]. The two parallel interfaces used are respectively the PVM/6000 and the PVMe [9] which are the IBM's implementation of the Oak Ridge pubblic-domain PVM [8].

Mesh Size	1	3	4
10×500	1	2.4	3.1
10×1000	1	2.8	3.5
50×500	1	2.9	3.7

Table (1). Parallel speedups of shell force evaluation kernel on four IBM RS/6000 mod. 560 connected by SOCC.

Mesh Size	1	3	4	6	8	15
10×500	1	2.9	3.8	5.4	6.7	9.6
10×1000	1	2.9	3.8	5.6	7.0	9.7
50×500	1	3.0	3.9	5.8	7.6	12.9
50×1000	1	3.0	3.9	5.8	7.6	13.3

Table (2). Parallel speedups of shell force evaluation kernel on 16-nodes SP1 connected by HPS.

Data reported in table 1 and 2 confirm the good level of parallel efficiency achieved even for moderate mesh size, giving a reasonable confidence that the parallel implementation used allows to efficiently exploit scalable parallel systems.

Preliminary tests on some realistic geometries, even accounting contribute of forces on three-dimensional elements and effect due to extended contact, confirm a comparable parallel efficiency. Some further efforts for improving parallel contact-impact algorithm are under progressing [5]. The strategy used wants to combine an efficient searching of elements, that will be in contact during the simulation, with the optimization of data communicated among processors.

Conclusions

We have reported about the basic parallelization strategy for the shell time-marching algorithm used in PAM-CRASH. The usage of a valid domain decomposition tool in conjunction with an efficient communication of data between adjacent subdomains allows to achieve a good parallel efficiency. The highly tuned message passing interface PVMe sensible improves communication bandwidth and latency on efficient networks which provides an higher parallel scalability even running on a larger number of powerful interconnected processors.

Acknowledgments

The first parallel implementation of PAM-CRASH based on PARMACS was carried out as part of the ESPRIT Project CAMAS (Project No. 6756).

References

[1] E. Haug, J. Clinkemaillie, J. Aberlenc, "Contact-Impact Problems for Crash" Second International Sympsium of Plasticity, Nagoya, Japan, August 4-5, 1989.

[2] R. Hempel, *The ANL/GMD Macros (PARMACS) in FORTRAN for Portable Parallel Programming using the Message Passing Programming Model*, Technical Report, GMD, St. Augustin, Germany, November 27, 1991.

[3] G. Lonsdale, J. Clinkemaillie, S. Vlachoutsis, J.F. de Ronde, P.M.A. Sloot, N. Floros and J. Reeve, "Crashworthiness Simulation Migration to Distributed Memory, MIMD Machines" presented at Conference on Supercomputing Applications in the Automotive Industries of the 26th ISATA, Aachen 13-17 September, 1993.

[4] G. Lonsdale, J. Clinkemaillie, S. Vlachoutsis and J. Dubois, "Communication Requirements in Parallel Crashworthiness Simulation", presented at HPCN Europe 1994 - The International Conference and Exhibition on High-Performance Computing and Networking, München, 18-20 April, 1994.

[5] G. Lonsdale, J. Clinkemaillie, S. Vlachoutsis and J. Dubois, "Crash-Simulation Migration to HPC Systems", presented at HPCN Europe 1994 - The International Conference and Exhibition on High-Performance Computing and Networking, München, 18-20 April, 1994.

[6] H.D Simon, *Computing Systems in Engineering*, **22**, PP.135-148, 1991.

[7] *IBM POWERparallel Technology Briefing: Interconnection Technologies for High-Performance Computing*, available from IBM branch office, December 1993; see also J. M. Kuzela, "IBM POWERparallel System - Sp2. Performance Mesaurements", available from IBM branch office, May 1994.

[8] *A User's Guide to PVM Parallel Virtual Machine*, Document n.ro ORNL/TM-12187, by the Oak Ridge National Laboratory, May 1993.

[9] *IBM AIX PVMe, User's Guide and Subroutine Reference, Release 3.0*, document n.ro SH23-0019-02, available from IBM branch office, 1994.