PARALLEL PROCESSING IN MSC/NASTRAN

Dr. Louis Komzsik
Chief Numerical Analyst

Numerical Methods Development Branch
The MacNeal-Schwendler Corporation
815 Colorado Blvd.
Los Angeles, CA 90041

Presented at
The 1993 MSC World Users' Conference
Arlington, Virginia

ABSTRACT

The MacNeal-Schwendler Corporation has been researching parallel computational methods and evaluating their applicability to its products since 1985. Limited parallelism has been offered in various MSC/NASTRAN products, mainly on supercomputers, since 1987. Presently, there are 5 major computer platforms where shared memory parallel execution is supported.

The paper will discuss some of the technical details of the shared memory parallel methodology and explore the limitations of parallel speedup on the current parallel environments using MSC/NASTRAN. Results of the leading parallel applications will be shown on a moderate number of processors, in specific a 4 fold parallel speedup on 16 CPUs analyzing a large automobile industry job will be demonstrated.

The results of investigations on using a distributed memory methodology (applied in the massively parallel computers) will also be discussed. The main strategic aspect of Lagrange multiplier based solution sequences and the topic of supermodules will be briefly addressed.

Introduction

The introduction of parallel processor supercomputers in the 1980's provided a possibility to increase application software performance. Some applications that are inherently parallel in nature, - such as image processing, atmospheric modeling and seismic processing, - were able to demonstrate significant performance improvements. On the other hand, the finite element application software area was not able to reach significant speedup. Sections 1 and 2 discuss the reasons for this.

The main distinction between different parallel processing computers is the organization of memory. The shared memory paradigm, which first appeared in the parallel supercomputers, has provided an easy application of parallelism to finite element software, at least in the numerical methods area. Techniques applied in this area, as well as MSC/NASTRAN results on shared memory parallel computers, is the topic of Section 3.

The distributed memory idea came along with the so-called hypercube supercomputers. These computers, which were first intended as inexpensive supercomputers, employed the technically simple distributed memory solutions; however, a significant price was paid in developing software. The cost of moving data around or even replicating information led to a gradual disappearance of the true hypercubes. Nevertheless, the distributed memory idea still serves as the technically feasible solution for computers with a large number of processors, the massively parallel computers. In Section 4, MSC's efforts and plans in this area are documented.

1. General Limits of Parallel Processing

An important result of the 1970's is the principle, which later became known as Amdahl's law, that the performance of an application on a supercomputer having two different execution modes (i.e., vector and scalar) is determined by the percentage of the slower portion. If a code, for instance, is 50 percent vector and 50 percent scalar, then the maximum improvement by vectorized execution can not exceed twofold, even if the vector operations are infinitely faster than the scalar operations. This results in a discouraging, hyperbolically decreasing speedup curve as a function of the serial percentage, shown in Figure 1.

AMDAHL'S LAW

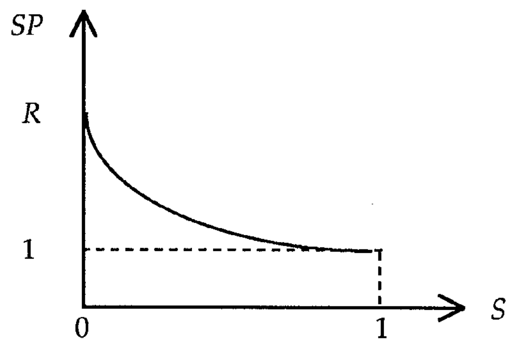

Figure 1.

The curve is described by the following mathematical expression:

$$SP = \frac{1}{S + \frac{1-S}{R}} \qquad (1)$$

where:

SP = Speedup

R = Vector/Scalar Speed Ratio

S = Scalar %

It was later recognized that Amdahl's law unfortunately also applies to parallel processing. If 90 percent of a code can be executed in parallel and 10 percent remains serial, the maximum speedup, even with infinite number of processors, can only be tenfold. A desperate effort to overcome this limitation resulted in the idea of scalability and the scaled speedup curve calculated by projecting a parallely executed program's performance back to the serial execution. This gives a more sympathetic, linearly decreasing speedup curve as function of the serial percentage shown in Figure 2. Some in the industry debated the validity of this approach, nevertheless it makes parallel processing more attractive.

SCALABLE SPEEDUP

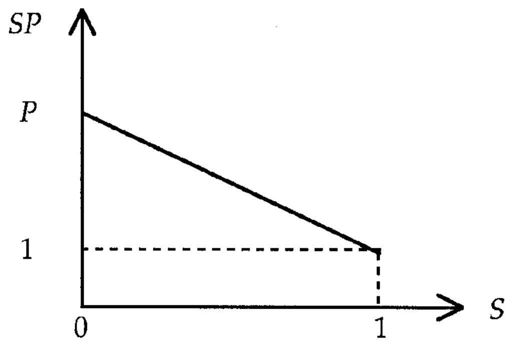

Figure 2.

The curve is described by:

$$SP = P + (1 - P)S \tag{2}$$

where:

SP = Speedup %

P = Number of Processors

S = Serial %

Unfortunately both curves imply that the speedup by vector or parallel processing is limited by the scalar or serial portion of the application software.

2. Limits of Parallel Processing in Finite Element Software

In order to understand the limits of parallel processing in finite element software, let us assume two simple uniformly meshed 2D and 3D models, a plate and a cube respectively. The cost associated with finite element analysis of these structures is dependent upon the degrees of freedom (DOF), i.e., the size of the finite element matrices and the average bandwidth.

In the case of a 2D n x n mesh (having n elements in one direction) the number of DOFs is $O(n^2)$ while the bandwidth is of (O)n. The corresponding numbers for a 3D mesh (n x n x n) are $O(n^3)$ and $O(n^2)$, respectively.

Let us assume that the serial time of a finite element analysis is proportional to the number of DOFs (N) and to a data move time (P):

$$T_s = N \cdot P \tag{3}$$

where *s* stands for serial. The parallelizable (potentially parallel) time is proportional to the number of DOFs and the bandwidth (B), as well as to a numerical kernel time (M):

$$T_p = N \cdot B^2 \cdot M \tag{4}$$

where *p* stands for parallel. The times resulting from substituting the different *N* and B values for 2D and 3D problems are in the following table:

	Serial	Parallel
2D	$T_{s,2D} = n^2 . P$	$T_{p,2D} = n^3 . M$
3D	$T_{s,3D} = n^3 . P$	$T_{p,3D} = n^7 . M$

Table 1.

A maximum speedup is calculated as the ratio of the parallel time to the serial time. This is based on Amdahl's law assuming that the parallel cost is eliminated in the parallel run and the serial cost is negligible in the serial run.

Using the formulae from Table 1 we get:

$$S = \frac{T_p}{T_S} = n^* \cdot \frac{M}{P} \tag{5}$$

where $*$ is $7/3$ for 3D and $3/2$ for 2D problems. This formula establishes a scalability rule of parallel finite element runs as follows: the speedup on a problem k times the size increases only by $k^{3/2}$ for 2D and $k^{7/3}$ for 3D meshes. For example, the speedup of a 2D problem twice the size of a baseline run increases only by about 2.8.

3. Shared Memory Parallel Processing in MSC/NASTRAN

Shared memory parallelization has been researched at MSC since 1985. We have had production versions of MSC/NASTRAN running in parallel since 1987, starting with Version 65. This shared memory paradigm is currently supported on certain CRAY, IBM, CONVEX, DEC and SGI computers.

In these computers, parallelism is exploited in MSC/NASTRAN numerical modules only. Currently, the parallel modules are: DCMP, FBS, MPYAD and REIGL. Several other modules having calls into these four modules could also benefit from parallel execution. The methods of invoking parallel execution are described in Reference 1 and the technical details of some of the algorithms are discussed in Reference 2.

The main solution sequences where parallel execution may help the performance are the various static and normal modes solutions, however, there is no engineering reason to restrict parallel methods to these solution types. The fact that only those solutions are using parallelism is merely due to implementation priorities.

We demonstrate the possible parallel processing speedup with a model of a car body of about one million degrees of freedom. The linear static analysis of this job was executed with MSC/NASTRAN Version 67.5 on a CRAY-YMP computer. The memory used was 75 million words and about 6 gigabytes of disk space was required for the execution.

The amount of I/O was over 16 gigabytes. Only the decomposition operation was executed in parallel. The timing data of the run is shown in Table 2.

# of CPUs	Elapsed Time (min:sec)	CPU Seconds	I/O Seconds
1	106:17	5,830.5	2,371.2
2	86:06	6,037.5	2,371.2
4	74:50	6,117.5	2,371.2

Table 2.

From Table 2 it is easy to see that we have a decent speedup of about 40% in elapsed time. The really good news is that the CPU overhead for parallel processing is below 5%. Considering all the limitations discussed in the earlier sections, these results are as good as one can get in a commercial finite element code using module level prallelism only. Finally there is no change in the I/O seconds, which is another important fact, demonstrating that we did not increase the amount of data created or moved during the run.

A further detailed analysis of this one million degree of freedom problem showed about 6% of the run is inherently serial. Approximately 84% is "potentially" parallel. That means that parallel execution is feasible for 84% of the work but has not necessarily been implemented in MSC/NASTRAN. The remaining 10% is due to overhead type operations related to database management and executive system tasks.

Of course, the above results change depending on the actual model, number of subcases, load vectors or eigenvectors requested. Nevertheless, the main conclusion is that if everything possible is made parallel in MSC/NASTRAN the maximum speedup of a huge job like this cannot exceed about sixfold.

Another large (several hundred thousand degrees of freedom) automobile industry job, ran on a 16 CPU CRAY C90, using all the currently available parallel modules in MSC/NASTRAN, was used to verify that fact.

Table 3 shows the details of this run.

# of CPUs	Elapsed Seconds	CPU Seconds	Speedup
1	3,441	3,043	--
2	2,179	3,209	1.57
4	1,366	3,250	2.52
8	1,054	3,270	3.26
16	859	3,319	4.01

Table 3.

This table shows that with our current code we can reach a speedup of about a factor of 4 by using 16 CPUs. The parallel overhead due to the larger number of CPUs is about 9%. That is actually an indication that the number of CPUs applied has approached its feasible limit. On the other hand we can also see that within the current small number of parallel modules we were able to reach a speedup of 4 out of the "ultimate" limit of 6 calculated earlier based on the million degrees of freedom problem.

4. MSC/NASTRAN on Distributed Memory Machines

We have been investigating the distributed memory or massively parallel machines since 1990. One of the conditions we had was that we would like to maintain the current MSC/NASTRAN framework as much as possible. We also identified the fact that increased pre- and post-processor support is needed. The 6% inherently serial time of the one million degrees of freedom run analyzed earlier was mainly in the beginning and at the end of the run in operations that can be moved into an interactive environment.

The possible parallel speedup we hope for is about 10-15 fold in the distributed memory version applying 16-64 CPUs. There are two main strategic aspects of the distributed memory version: one is the application of a Lagrange multiplier method for constraint processing, and the other one is a supermodule approach.

The Lagrange multiplier approach for constraint processing is well known in the industry. Its main advantage is to eliminate some of the constraint elimination operations. These operations, in the SCE1, MCE1 and MCE2 modules, have high serial content and therefore inhibiters of further parallel speedup. In this method we create an augmented system of equations, in a sense handling the constraints as elements. DMAP alters for this method were delivered to users with MSC/NASTRAN Version 67.5 and will gradually migrate into mainstream solution sequences.

Another technical aspect of the distributed version is a so-called supermodule approach. Besides moving some pre and post-solution operations into the pre- and postprocessor product, we plan to combine several modules into supermodules. This will minimize sequential operations and data transfer, both crucial in parallel processing. The price of this will be paid in some restrictions in user interactions, such as alters and restarts.

The manpower and cost requirements of the distributed parallel version of MSC/NASTRAN are much higher than the cost associated with shared memory parallelism. Due to this, it is immature to say which version of MSC/NASTRAN will carry this capability.

Conclusion

We believe we have an adequate capability to take advantage of shared memory parallelism in current MSC/NASTRAN versions. Considering the theoretical limitations, the results demonstrated in the paper are good; and we will continue work in this area.

We found that it is not worth pursuing much more in the shared memory parallel environment because the cost of reaching the remaining 1/3 of possible parallel speedup is not proportional to the gain. There is more to be gained by a distributed memory version of the code.

We have established a corporate strategy to run on distributed memory (massively parallel) computers in the future. Preliminary development work is already underway.

The main goal of our effort in both areas is to provide parallel capabilities without limiting our leading characteristics in overall performance, numerical accuracy or reliability. In order to achieve this goal we are continuing to research on advanced concepts related to these topics.

References

[1] Komzsik, L., MSC/NASTRAN User's Guide for Numerical Methods, The MacNeal-Schwendler Corporation, Los Angeles, CA, November, 1992.

[2] Komzsik, L., MSC/NASTRAN Handbook for Numerical Methods, The MacNeal-Schwendler Corporation, Los Angeles, CA, November, 1990.

[3] Goehlich, D., Fulton, R., and Komzsik, L., "Application of a Parallel Equation Solver to Static FEM Problems," International J. for Computers and Structures, Vol. 31, No. 2, 1989.

[4] Goehlich, D., Fulton, R., and Komzsik, L., "Program Designs and Data Structures for Vectorized Finite Element Computations," Vol.2: Supercomputer Applications, Proceedings, Fourth International Conference on Supercomputing, edited by L. P. Kartashev and S. I. Kartashev, International Supercomputing Institute, Inc., St. Petersburg, FL, 1989, pp. 415-420.

[5] Goehlich, D., and Komzsik, L., "Decomposition of Finite Element Matrices on Parallel Computers," Proceedings of the ASME International Computers in Engineering Conference, American Society of Mechanical Engineers, New York, 1987.

[6] Komzsik, L., "Parallel Static Solution in Finite Element Analysis," Vol. 2: Industrial Supercomputer Applications and Computations, Proc., Second International Conference on Supercomputing, edited by L. P. Kartashev and S. I. Kartashev, International Supercomputing Institute, Inc., St. Petersburg, FL, 1987, pp. 138-144.

[7] Komzsik, L., "MSC/NASTRAN on Minisupercomputers," Proceedings, MSC European Users Conference, Rome, Italy, 1988.

[8] Komzsik, L., "The Evolution of Eigenvalue Methods and Computer Hardware," Invited presentation IBM European Institute, Oberlech, Austria, 1988.

[9] Komzsik, L., "On the Adaptation of MSC/NASTRAN to Supercomputers," Proceedings, International Conference on Supercomputing, Pisa, Italy, 1989.

[10] Komzsik, L., "Vibration Analysis of Bus Structures Using MSC/NASTRAN," Proceedings, International Conference of Bus and Coach Experts, Budapest, Hungary, 1990.

[11] Komzsik, L., "Optimization of Finite Element Software Systems on Supercomputers," Supercomputing in Engineering Analysis, edited by H. Adeli, Marcel Dekker, 1991.

[12] Komzsik, L., "New Feature of the Lanczos Module in Version 67 of MSC/NASTRAN," Proceedings, MSC European Users Conference, Chemsee, Germany, 1991.

[13] Komzsik, L., and Rose, T., "Substructuring in MSC/NASTRAN for Large Scale Parallel Applications," Proceedings, NASA-USAF Symposium on Parallel Methods on Large-Scale Structural Analysis and Physics Applications, NASA Langley Research Center, 1991 and in Computing Systems in Engineering, Pergamon Press, Vol. 2, Number 1, 1991.

[14] Komzsik, L., "The Past, Present and Future of Supercomputing," Keynote Lecture, 5th Australian Supercomputing Conference, Melbourne, Australia, 1992.

[15] Komzsik, L., and Chiang, K., "The Effect of a Lagrange Multiplier Approach in MSC/NASTRAN on Large Scale Parallel Applictions," Symposium on Parallel Methods for Large Scale Analysis and Design, NASA Langley Research Center, 1993.

[16] Komzsik, L., "History and Trends of Supercomputing," Keynote Lecture, Supercomputing '93, Utrecht, The Netherlands, 1993.