Parallel Computing and Performance Study of Vehicle Crash Safety Numerical Simulation

QI Wenguo, JIN Xianlong, ZHANG Xiaoyun

High Performance Computing Center

Shanghai Jiao tong University, Shanghai 200030, China

Abstract – Parallel computing is efficient solutions to speed up and enhance the solving ability of large-scale numerical simulations, such as vehicle crash safety simulation based on the nonlinear finite element method. In this paper, a cost-effective domain decomposition method based on contact balance is presented，and the algorithm flowchart including contact computing is drawn, and the parallel computing process and communication overhead are analyzed. Furthermore, scalability of parallel computing method on different hardware platforms is studied. At last, effect of different domain decomposition strategy on automobile crash simulation computing efficiency is presented. To end users, research result is a guide to choose appropriate hardware platform and computing software.

Keywords: Parallel Computing；Crash；Finite Element Method；Domain Decomposition；Contact

1 Introduction

Vehicle traffic accidents bring dramatic tragedy to people and are severe social problems. In the area of vehicle crash safety researches, Finite element simulation is an effective and low-cost method which improves the crashworthiness and is propitious to reduce the R&D cost and shorten the design time in the developing stage of vehicle design. The vehicle crash simulation started 17 years ago with the first successful VW Polo frontal crash overnight simulation on a CRAY-1 computer [1]. Vehicle crashworthiness research using the finite element method is widely used today [2-4].

Vehicle impact is a complicated nonlinear dynamic contact process because automobile structures experience dramatic impact loads in very short time (usually in 100ms). During the impact process, automobile structures usually experience buckling deformation including physical nonlinearities, geometric nonlinearities and material nonlinearities. Detailed finite element vehicle models are in the range of 200000-500000 elements to obtain more accurate and robust simulation results, which need more and more CPUs and a lot of storage resources, such as memory and hard disk. But the accurate results are difficult to obtain based on the fact that traditional finite element codes only depend on single processor. With the rapid development of computer hardware, commercial finite element codes begin to use multiple processors in parallel to solve the large-scale nonlinear finite element analysis by virtue of parallel computing technology.

In this paper, the domain decomposition method for parallel computing of vehicle crash safety simulation is introduced simply. The algorithm flowchart including contact computing is drawn, and the parallel computing process and primary overhead are analyzed. Furthermore, scalability of parallel computing method on different hardware platforms is studied. At last, effect of different domain decomposition strategy on automobile crash computing efficiency is presented. To end users, research results are a guide to choose appropriate parallel computing platforms including hardware and

Contact E-mail: wenguoqi@sjtu.edu.cn

computing software.

2 Numerical Simulation and Parallel Strategy of Vehicle Crash Safety

2.1 Nonlinear Finite Element Method

Vehicle crash simulation may use explicit and implicit method. There are several advantages to use the explicit method: ①it is not to build and solve the stiffness matrix. ②it is easy to deal with contact process because time step is enough small. At present, vehicle crash numerical simulation mainly adopts explicit time integral method [5].

The whole vehicle is meshed into many elements, according to the strain-displacement relation and stress-strain relation, the dynamic equilibrium equation at time t is listed as follows:

$$M\ddot{u}_t = Q_t - F_t + F_{ct} \tag{1}$$

where each dot on top of a variable refers to differentiation with respect to time, M is the mass matrix; u_t is displacement vector; Q_t and F_t account for external applied loads and internal resistance forces, respectively; F_{ct} is contact force.

To solve equation (1), the central difference method is adopted usually. To advance to time $t+\Delta t$, the central difference equations can be given:

$$\dot{u}_{t+(\Delta t/2)} = \dot{u}_{t-(\Delta t/2)} + \Delta t \cdot \ddot{u}_t \tag{2}$$

and

$$u_{t+\Delta t} = u_t + \Delta t \cdot \dot{u}_{t+(\Delta t/2)} \tag{3}$$

Thus, according to equation (2) and (3), the acceleration at time t may be written

$$\ddot{u}_t = \frac{u_{t+\Delta t} - 2u_t + u_{t-\Delta t}}{(\Delta t)^2} \tag{4}$$

according to equation (1) and (4), the displacement at time $t+\Delta t$ may be written

$$u_{t+\Delta t} = M^{-1}[(\Delta t)^2(Q_t - F_t + F_{ct}) + 2Mu_t - Mu_{t-\Delta t}] \tag{5}$$

where Δt is time step. $t+\Delta t$ and $t-\Delta t$ refers to next and previous time increments, respectively. The elemental mass is lumped at the nodes, so that the mass matrix M is diagonal. Then equation (5) can be disassembled 5N equations (N is total node amount of finite element model) . Parallel finite element method is to solve equation (5) in parallel. At present, the most useful method is the domain decomposition method [6].

2.2 Strategy and Principle of the Domain Decomposition Method

The domain decomposition method is first to divide complex system or structure into several subsystem or sub-domain according to some principles such as physical characters、geometrical shape and element variety. Next, each subdomain is assigned to different processor in order to realize parallel computing. Interprocessor communication is made with massage passing libraries such as Message Passing Interface (MPI) or Parallel Virtual Machine (PVM). Theory analysis and practical computing demonstrate that the domain decomposition method can provide highly parallel and scalable algorithm for large-scale nonlinear finite element analysis [7].

In the vehicle crash simulation, the authors make use of the nonlinear finite element method based on the domain decomposition method. First, the authors set up the finite element model of the auto-body and define initialize condition、boundary condition、material models、contact condition、 result output format, etc. Next, the partitions for parallel computing are defined by a variety of codes available on the worldwide web, such as Metis, or by hand. Thus the whole finite element model is partitioned several subdomains. Figure 1 shows the partition result. At last, each subdomain is assigned to different processor during one time step, every processor solves the equations which belong to corresponding subdomain. The computing result includes displacement, velocity, acceleration and stress, etc. Interprocessor communication only needs transmitting the displacements of interface elements at previous time increments.

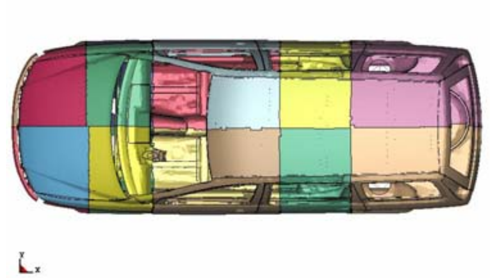

Fig.1 Domain decomposition model of the vehicle (16 subdomains by RCB)

A coarse-grained parallel computing can be got by the domain decomposition method because the explicit finite element method is employed at each element nodes.

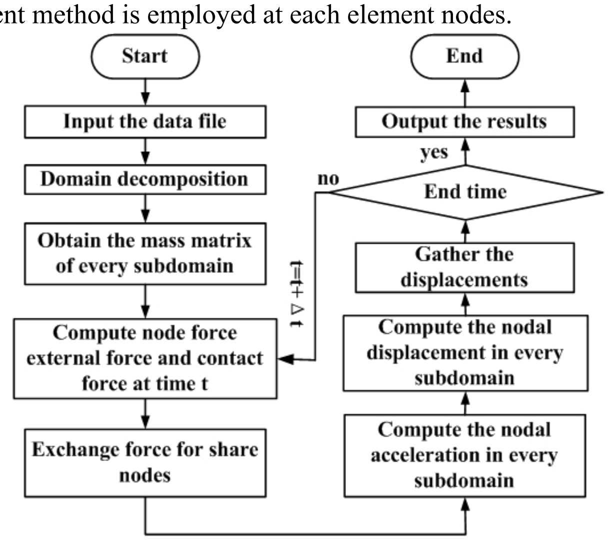

Fig.2 Flowchart of explicit FEM parallel computing

As shown in Figure 2, to implement a parallel algorithm, it is necessary to discretize the given domain into subdomains when the data file is inputted, and analyze each subdomain by a processor. The results are then assembled for the overall solution.

An efficient domain decomposition algorithm should satisfy the following properties [8]:

1. It should be able to handle any irregular mesh geometry.
2. The number of interface nodes or elements between subdomains should be minimized. This helps to reduce the size of the overall problem.
3. The subdomains should contain an approximately equal number of elements in order tobalance the computational load between processors.

At present, the most useful domain decomposition method is the recursive coordinate bisection algorithm, that is, RCB. The basic principles are listed below:

1. Judge the model length in the three coordinate axes, and divide the model in half perpendicularly to the longest coordinate axis.
2. Judge the current piece of model in the three coordinate axes, and divide the model in half perpendicularly to the longest coordinate axis.
3. According to the strategy, repeatedly dividing a domain into two subdomains until the algorithm satisfies the converge conditions, such as the number of processors used in the simulation.
4. Output the results.

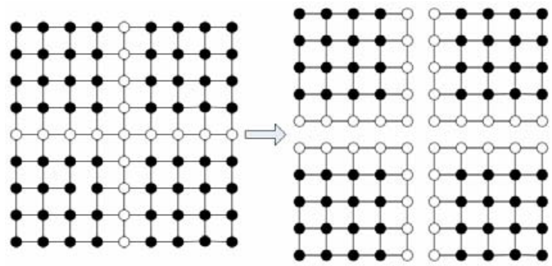

Fig.3 Description of recursive coordinate bisection algorithm

As shown in Figure 3, the mesh is usually divided into subdomains whose shapes are rectangle or cube along the coordinate axis, which is decided by the characteristic of the algorithm. The RCB algorithm may insure each subdomain has an approximately equal number of elements, but it only depends on the geometry information of the numerical model and ignores the load distributions and contact locations.

3 Contact Balance Bisection Algorithm for Domain Decomposition

Contact plays an important role in the aspect of computation accuracy and cost in the crash safety simulation of vehicle structures. Contact calculations can be extremely computationally intensive. Time spent in contact calculation (for example, contact searching and contact force computing) has usually 40 percent of the total elapsed time, even 60 percent. Researches on the parallel strategy for the contact calculations are important [9].

Subject to the characteristics of contact-impact problems, a new domain decomposition method named the contact balance bisection algorithm, namely CBB, is designed based on the recursive coordinate bisection algorithm. The principles are:

1. Find out the elements associated with contact interfaces
2. Divide these contact elements evenly according to the number of processors, and get the boundary of subdomains
3. Decompose the whole domain according to the boundaries, and the nodes on the boundaries are duplicated, and then distributed to corresponding subdomains.
4. Divide the remainder elements according to the geometry coordinate until the algorithm satisfies the converge conditions.
5. Output the results

There are two factors to affect speedup in parallel computing: ①overhead due to load imbalance. Because the subdomains are established incorrectly, load imbalance among the processors means that some processors will complete their portion of the work and must wait for the others. ② communication overhead among the processors [10]. The formula is: $T(m,n)=t_0(n)+m\cdot t_c(n)$. m is the length of message; n is the number of processors; $t_0(n)$ is startup time, which varies with different computer platforms; $t_c(n)$ is communication time per byte, which depends on the architecture of computer system. For a given parallel platform, the amount and cost of communication among the processors increase as the number of CPUs increases.

During every time step, each processor advances the solution for its own subdomain to the end of that time step. This process is independent of all other subdomains, so it is highly parallel. However, before work on the next time step can begin, communication must occur to relate information on the state of the solution to neighboring subdomains. Once this communication is complete, the solution phase of the next time step begins. The shorter elapsed time of a time step solution can be obtained as more processors are used for the simulation, but more subdomains would lead to huge communication overhead, which influence the computing efficiency, so it is important to select appropriate number of processor used for the simulation based on the number of elements in the model.

4 Parallel Computing of Vehicle Crash Simulation

The vehicle model consists of more than 120 parts, and the numerical model involves approximately 170000 elements. Figure 4 shows the finite element model of the vehicle. According to National Crash Standard, CMVDR294, the frontal crash simulation is realized. During the simulation, the impact velocity is 48.3km/h.and the simulation time in this study is 100 milliseconds.

Tab.1 Type and number of element

	Part	Beam element	Shell element	Solid element
No.	127	40	166143	16

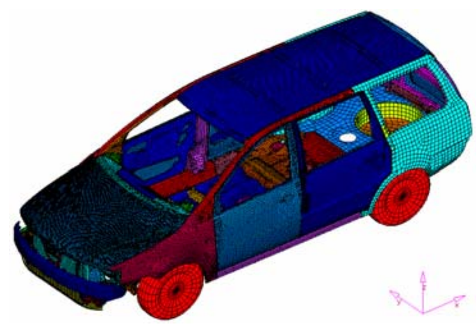

Fig.4 Finite element model of the vehicle

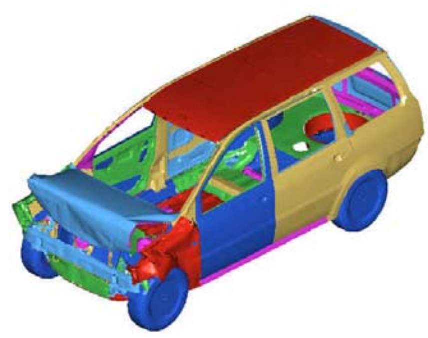

Fig.5 Simulation result of vehicle crash safety t=90ms

4.1 Scalability of Different Hardware Platforms

In order to research the computing efficiency of different hardware architectures, a series of vehicle crash simulations is made on two parallel computing platforms at Shanghai Supercomputer

Center. Platform 1, the SGI Onxy 3800, is a shared-memory massive parallel processors. The platform 2, ShenWei cluster, is cluster of workstations, which includes a number of workstations or PCs. The cluster has become the mainstream of parallel computing since 1990's because it is flexible, easy to construct and low cost [11]. This cluster of workstations adopted in this study consists of 8 slave nodes and 1 master node, each with a pair of 2GHz Intel P4 Xeon processors. All analyses described herein are made with dedicated processors and the highest performance communication channels available on each system.

Hardware and software of two platforms are listed in Table 2. On the platform 1, the vehicle frontal crash simulation is achieved using 1、2、4、8、16、32、64 CPUs respectively. Table 3 shows the simulation elapsed time and speedup.

Tab.2 Hardware and software of two platforms

Platform	SGI Onxy 3800	ShenWei
Number of CPUs	64	18
Type of processor	MIPS R10000	Intel P4 Xeon
Processor speed	500MHz	2GHz
Memory	32 G	18 G
Hard disk	4.3T	720G
Peak performance	64GFLOPS	36GFLOPS
Computing program	LS-DYNA 960	LS-DYNA 960
Operating system	SGI Irix6.5	Linux RedHat7.3
Parallel environment	MPI1.2.4	MPI1.2.4

On the platform 2, the vehicle frontal crash simulation is achieved using two projects: single processor per node and two processors per node respectively. Table 4 shows the total elapsed time and speedup.

Tab.3 Computing time and speedup of platform 1

NCPU	Elapsed time/s	Speedup
1	156539	1.00
2	84615	1.85
4	57131	2.74
8	35943	4.35
16	31183	5.02
32	18725	8.36
64	25485	6.14

Tab.4 Computing time and speedup of platform 2

Project 1: single processor per node			Project 2: two processors per node		
NCPU	Elapsed time/s	Speedup	NCPU	Elapsed time/s	Speedup

1	107491	1.00	1	107491	1.00
2	53042	2.03	2	60051	1.79
4	26855	4.00	4	33696	3.19
8	14957	7.19	8	18957	5.67
			16	11996	8.96

As depicted in Table 3-4, one may see that on a single processor, the total elapsed time of SGI Onxy 3800 is longer than ShenWei cluster. Because the numerical model and the amount of memory in analysis is identical, the low clock frequency of processor on SGI leads to the long elapsed time

The simulation elapsed time decreases rapidly as more processors are used for the analysis. For the vehicle model used in this study, the elapsed time actually increased as the number of processors went from 32 to 64. This number of elements in the vehicle finite element model generate too much communication overhead which can't be overcome with parallelism at these processor counts, and which leads to the parallel performance diminishes as the number of elements per processor becomes small.

Data for all simulations in the study is summarized in Figure 6. In this spot, the vertical axis represents the elapsed time speedup which occurs as the number of processors is doubled. As more and more processors are used, the gain form parallelism decreases. When using four or eight processors, the speedup of project 1 on platform 2 is better than the others. As shown in Figure 6, project 1 speedup is near-linear scaling because the partitioning is realized according to the number of computing nodes used in the crash simulation. For project 1, the number of subdomains is same as the processors; but the number of subdomains is only half as the processors in project 2. From Table 4, we can see that the elapsed time using 8 processors in project 2 is close to the time using 4 processors in project 1.

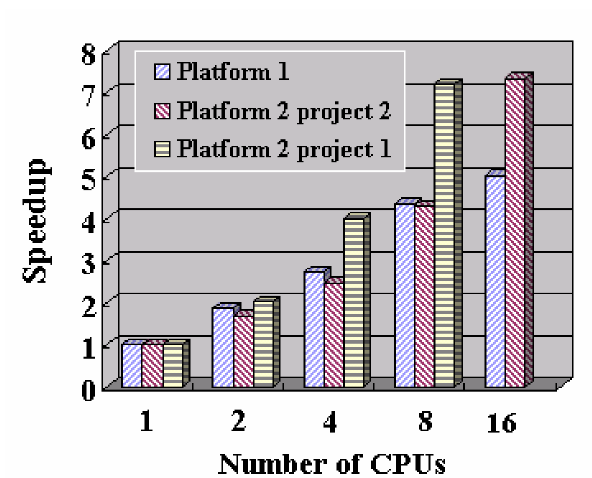

Fig.6 Speedup of different computing platform

4.2 Comparison of Different Domain Decomposition Algorithm

During vehicle front crash, most of the contact searching will take place in the front of the vehicle. The domain decomposition by RCB is displayed in Figure 1, where the algorithm divides the total number of elements into several subdomains with the same number of elements, but the partitioning is less than ideal because the subdomains at the front of the vehicle will require more computation than those at the rear of the vehicle, leading to load imbalance. A better domain decomposition by CBB is displayed in Figure 7, where most domains runs from the front of the vehicle to the back, so that all domains have similar contact characteristics, and better overall load balancing. According to the presented method, each domain has not only the same number of elements, but also an equal amount of computing load at each time step.

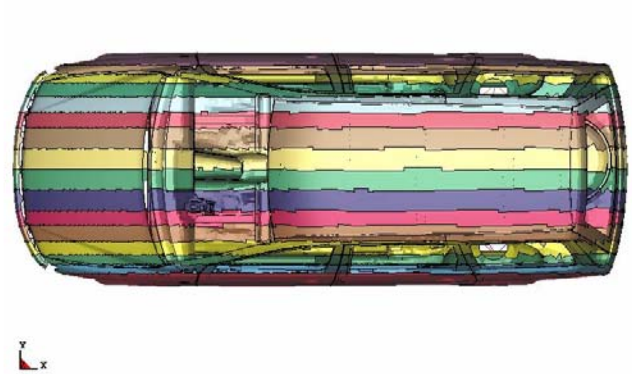

Fig.7 16 subdomains by CBB

Tab.5 Computing time and speedup of different domain decomposition algorithm

NCPU	RCB		CBB	
	Elapsed time/s	Speedup	Elapsed time/s	Speedup
1	107491	1.00	107491	1.00
2	60051	1.79	56278	1.91
4	33696	3.19	31157	3.45
8	18957	5.67	18056	5.95
16	11996	8.96	10728	10.02

The results in Table 5 and Figure 8 illustrate that using the same number of CPUs, the parallel efficiency of CBB is better than RCB, and the performance improves more significantly with 16 CPUs specially.

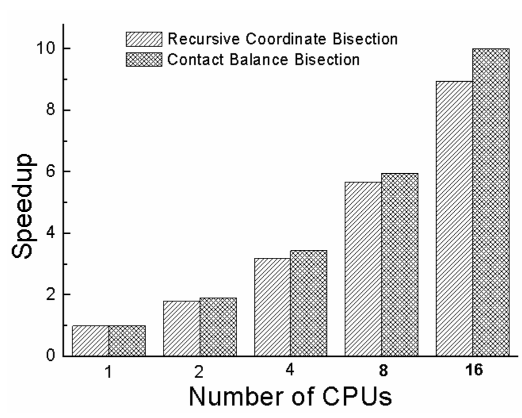

Fig 8 Speedup of different domain decomposition algorithm

5 Conclusions

Through aforementioned researches, some conclusions can be drawn:

1) For vehicle crash safety simulation using a single processor, the simulation elapsed time may depend on the clock frequency of processor.

2) It is very important to select appropriate number of processor used for the simulation based on the number of elements in the model. In this study, 20,000 element crash simulations may need 8 or 16 processors.
3) For vehicle crash safety simulation using up to 8 processors, partitioning with RCB can provide good performance. For 16 or more processors, CBB partitioning provides better performance than RCB.
4) Cluster of workstation is a more attractive and economical way in parallel computing from the consideration of hardware cost and computing performance.

6 Acknowledgements

The authors gratefully acknowledge the financial support from the National Natural Science Foundation of China (No.60174023) and the Ph.D.Programs Foundation of Ministry of Education of China (No.20010248008). The authors appreciate the assistance and suggestions of many individuals associated with the Shanghai Supercomputer Center and with SJTU High Performance Computing center.

7 References

1. Eberhard H, Clinckemaillie J, Xiaomin N. Recent trends and advances in crash simulation and design of vehicles [J]. Chinese Journal of Mechanical Engineering, 1998, 34(1): 93-99.
2. Zhang Wei-gang, Zhong Zhi-hua. Crashworthiness improvement of passenger cars with computer simulation techniques [J]. Chinese Journal of Mechanical Engineering, 2002, 38(3): 135-138.
3. Chweizerhof K, Nilsson L, Hallquist J O. Crashworthiness analysis in the automotive industry [J]. International Journal of Computer Applications in Technology, 1992, 5(4): 134-156.
4. Zhong Zhi-hua. Finite element method in crashworthiness analysis of automobiles [J]. Automotive Engineering, 1994, 16(1): 1-6.
5. Zhong Zhi-hua. Finite element procedures for comtact-impact problems [M]. Oxford University Press, 1993.
6. Wang Fu-jun, Cheng Jian-gang, Yao Zhen-han. Parallel algorithm for explicit integration method in nonlinear dynamic structural analysis [J]. Journal of Tsinghua University (Science and Technology), 2002, 42(4): 431-434.
7. Rama Mohan Rao, T.Appa Rao,B.Dattaguru. A new parallel overlapped domain decomposition method for nonlinear dynamic finite element analysis [J]. Computers & structures, 2003, 81: 2441-2454.
8. kaveh, H.A.Rahimi Bondarabady. A hybrid graph-generic method for domain decomposition [J]. Finite elements in analysis and design, 2003, 39: 1237-1347.
9. Kou Zhe-jun, Cheng Jian-gang, Yao Zhen-han. Study of scalable parallel computation for compact-impact problems [J].Chinese Journal of Computation Mechanics, 2003, 20(3): 325-328.
10. Hwang Kai, Xu Zhi-wei. Scalable Parallel Computing: Technology、Architecture, Programming [M]. Beijing: China Mechanical Industry Press, 2000.
11. Kou Zhe-jun, Cheng Jian-gang, Yao Zhen-han. Simulation of automobile crashworthiness using clustered parallel computers [J]. Journal of Tsinghua University (Science and Technology), 2003, 43(5): 666-672.