Parallel Computing of Numerical Simulation in Building Fires

YAO Hao-wei, ZHAO Zhe, LIANG Dong
College of Engineering, Sun Yat-sen University, Guangzhou, China
Email: yaohaowei@gmail.com

LIU Song-yang
Department of Electronic Information, Zhuhai College of Jilin University, Zhuhai, China
Email: lsysea@126.com

ZHANG Liang
Architecture Design and Research Institute, South China University of Technology, Guangzhou, China
Email: 091278@163.com

***Abstract*—Fire Dynamics Simulator parallel computing program has been compiled based on Message Passing Interface. The fire flow field is divided into several segments and each segment is allocated to the corresponding node to calculate. The calculation shows that computing time has been significantly reduced as the amount of processors increase. The procedure and method that the text has established is feasible, and could be further extended to large-scale parallel computing and engineering application.**

***Index Terms*—building fire, Computational Fluid Dynamics, Fire Dynamics Simulator, parallel processing systems, Message Passing Interface, acceleration ratio**

I. Introduction

During the design and assessment of the performance-based fire, it usually requires the use of fire modes to simulate the possible scenarios in the building fire. Among these fire modes, field simulation was widely used. It uses computers to solve the spatial distribution of the state parameters and its changes with time in the fire process, which can show the details of change in the fire process.

Fire Dynamics Simulator (FDS) is a computational fluid dynamics simulation software developed by National Institute of Standards and Technology (NIST), aiming at studying the movement of fluids in a fire. However, in the actual FDS calculation, general stand-alone configuration usually runs out of memory and slows down the calculation. Larger projects often take several weeks or even more days. Meanwhile, a considerable part of the research agencies can not be equipped with high-performance computers such as workstations because of some limitations. Therefore, it is necessary to introduce parallel computing to accelerate the speed of calculation.

A. Introduction to Parallel Computing

Parallel computing refers to the processing of multiple tasks, multiple instructions or multiple data simultaneously. The computer system to complete such a process is called a parallel computer system, which organizes multiple processors connected through the network in a certain way orderly.

The current CPU parallel technologies include non-symmetric multi-processor (AMY) technology, symmetric multi-processing (SMP) technology, cluster (Cluster) technology, NUMA distributed memory access technology, massively parallel processing (MMP) technology, etc. Symmetric multi-processing (SMP) technology could ensure that all the CPUs share the resource of the system so that the workload can be evenly distributed to each processor. As a result, it is used in most engineering calculations.

The parallel computing of the FDS software introduced in the paper uses symmetric multi- processing technology. To build up a SMP system, the most critical point is the match of the CPUs. Therefore, the product type, the operating frequency and the type of CPU should match to form the SMP systems [2], so that the computing platform can play the best performance. The FDS parallel computing is used to calculate the large-scale computational problems with multi-processors, while the principle is to divide the whole region to use parallel computing. The basic idea is as following Figure 1.

doi:10.4304/jcp.7.11.2680-2683

B. The Design and Build Hardware Platform

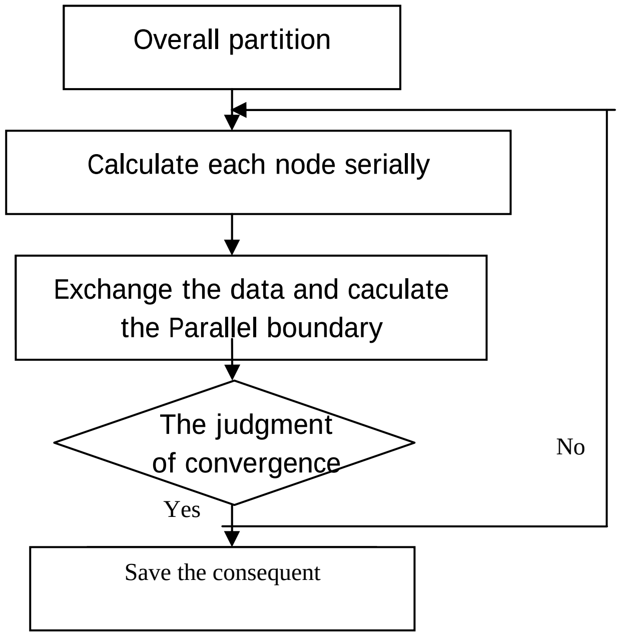

Figure 1. Flow chart of parallel computing

Under the conditions of the sureness of physical models and numerical algorithms, the speed of calculation depends on the number and the performance of CPU, the memory of CPU, the memory access bandwidth, the interconnect bandwidth nodes as well as the mesh quality and partition quality. Each specific problem or each particular machine corresponds to a best number of partitions. If the numbers of partitions are too many, the load of communication between CPUs would increase. And it would reduce the computing speed when the numbers of partitions increase to a certain extent. However, if the numbers of partitions are too small, it would also affect the speed of calculation because numbers of CPUs involving in calculation are inadequate [3].

The motherboard bandwidth under the structure of symmetric multi-processing could limit the speed of multi-CPUs reading the shared memory, which leads to the limitation of the communication speed between the nodes on the bus. Taking the above factors into account, this NIC uses the third generation of I/O bus technology, PCI-E-Gigabit Ethernet Gigabit Ethernet, the switch is the D-link DES-1016D/1000MHz, and the communication protocol between the master and slave servers adopts the RSH protocols.

The CPUs involved in the calculation of the paper are as following: Intel Core™ 2 E7400, clocked at 2.8GHZ; main server memory 8GB, slave-machine memory 4GB, which basically meet the need of computing requirements.

At the same time, the following two points should be paid attention in the process of the platform building.

1) The main server memory should be large enough (larger than the amount of data generated of a parallel computing).

2) Computing nodes in the machine performance should be uniform, to avoid the slowest node becoming bottlenecks.

Figure 2 shows the schematic diagram of the parallel computing platform. The CPUs perform in unanimity and communicate with each other through the switches.

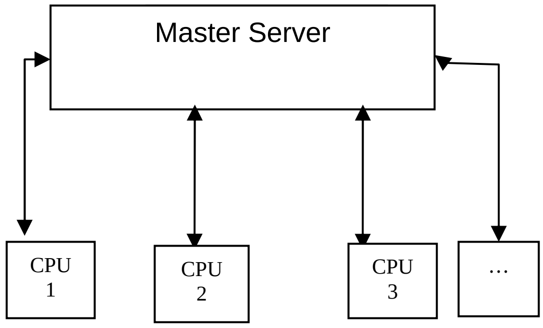

Figure2. Schematic diagram of parallel computing platform

II. THE ESTABLISHMENT OF THE SOFTWARE PLATFORM

This experiment uses 64-bit Windows System. Applications can pre-load enough data to the physical memory. So that the processors can access the data quickly. This feature reduces the time of loading the data into virtual memory, researching and reading the data from low-speed hard disk and writing the data into it, which will enable the application to run faster and more efficiently. In this experiment, six machines are set up to form a parallel processing system, named fds1、fds2、fds3、fds4、fds5、fds6 in order. The main server is fds1, and the rest the slave-machines. One thing should be noticed is that, when building the parallel computing environment, the project team didn't set the shared directory of the host machine as writable. This resulted in several days' meaningless work of the project team to eliminate errors. This could be a reference for others who undertake a relevant work.

A. MPI and MPICH

Message Passing Interface (MPI) is a norm of Message-passing parallel programming model, a standard which has no relationship with platform and language. The widely used standard in the current is MPI v1.2 and MPI v2.0. MPI parallel programming platform is constituted by the standard message-passing functions and related support functions. Multiple processes communicate by calling these functions. An order starts multiple processes at the same time to form multiple independent processes, running on different processors and owning separate memory space. The communication between processes is achieved by calling MPI functions.

MPICH is an MPI implementation with the greatest influence, which is widely used in various systems to support for parallel and distributed programming. It is developed simultaneously with the MPI specification, For instance, the MPI v2.0 is corresponded to the implementations version of MPICH2. This paper adopts MPICH2 to implement the data transfer between CPUs to achieve the purpose of improving significantly the parallel efficiency. Here MPI is installed on each of the computers within the network that will be used for FDS computations.

B. The Analysis of Parallel Computing Platform

Parallel computing mainly concerns the speedup, Parallel efficiency and scalability. Speedup refers that parallel computing accelerates the ability of the entire calculation process. Parallel efficiency means the average utilization of each processor. Scalability is the ability that parallel computing performance improves in proportion with the increase of the number of processors. Based on other efficiency measures, with the increase in the number of processors, if the increase of a smaller size of calculations could make the parallel computational efficiency remain unchanged, the parallel algorithm has good scalability.

Speedup:

$$S_n = T_1 / T_n$$

Parallel efficiency:

$$E_n = S_n/n * 100\%$$

Note:

T1 is the time required for a single processor.

Tn is the time that the n- processors parallel computing required.

Sn is the speedup

En is the parallel efficiency

The picture below is a single-room building, which would be taken as a fire example for FDS calculation. One of the length*width*height of the model is 1.5m*1 m*2m (for 3 million grids), the other is 3m*1m*2 m(for 6 million grids), so both of the grid size is 0.01m. The simulation time is 600s, and the ambient temperature is 20℃. The fire is located in the middle of the room, and the specified Heat Release Rate is 500KW.

The building model is shown in Figure 3.

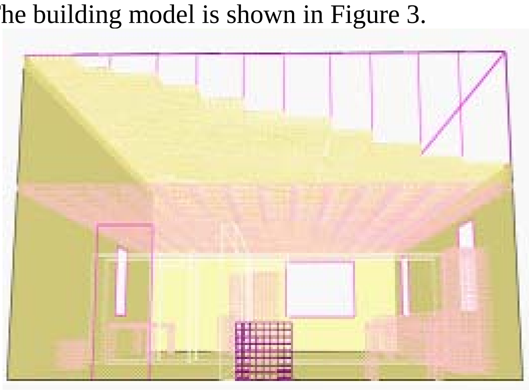

Figure3. Outline of the room FDS model

The calculation grid number is 3 million and 6 million respectively (the two numbers both fit for the FFTs) [6]. The grid is assigned to each CPU equally. Each experiment has been done for three times to get the average value. The results are shown in Table 1, Table 2.

TABLE 1
PARALLEL COMPUTATION OF A SINGLE ROOM FIRE TEST RECORD FORM (3 MILLION GRIDS, HERE NODES REFERS TO CPU NUMBER)

Nodes	Single-step iteration time / s	Speedup Sn	Parallel efficiency En (%)
1	12.63	1	100
2	6.58	1.92	96
3	5.15	2.45	82
4	3.77	3.36	83.7
5	3.11	4.04	80.8
6	2.67	4.73	79.3

TABLE 2
PARALLEL COMPUTATION OF A SINGLE ROOM FIRE TEST RECORD TABLE (6 MILLION GRIDS)

Nodes	Single-step iteration time /	Speedup Sn	Parallel efficiency En (%)
1	22.36	1	100
2	11.75	1.90	95
3	9.18	2.41	81
4	7.12	3.12	78.5
5	5.63	3.90	79.0
6	4.81	4.63	77.1

Single-step iteration time (td), shown in Figure 4, is reduced as processors increase under the condition of that the calculation scale keeps unchanged., which means that the calculation time required will be greatly reduced. When the number of nodes is 6, single-step iteration time is much less compared with that of only one.

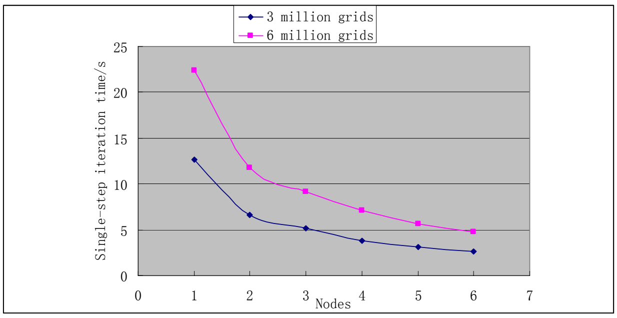

Figure4. Relationship between the number processor and iteration time step

Speedup (Sn), shown in Figure 5, grows as the numbers of processors increase, It can be seen from Figure 5 that the speedup of the smaller scale is larger than that of the larger scale.

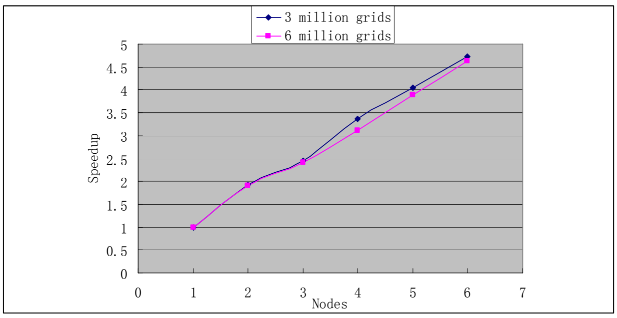

Figure5. Relationship between acceleration ratio and the number processor

Parallel efficiency (E), shown in Figure 6, keeps almost unchanged as the grid size increases, which has been revealed by comparing the 3 million grid with the 6 million one. However, if the number of grids is unchanged, the parallel efficiency decreases as the number of processors increase. The reason is that when the numbers of processors increase, the data exchange between regions also increases. And traffic is relatively larger, which makes the parallel efficiency reduce. Then a prediction comes out that when the numbers of processors are too many, parallel efficiency will be significantly reduced.

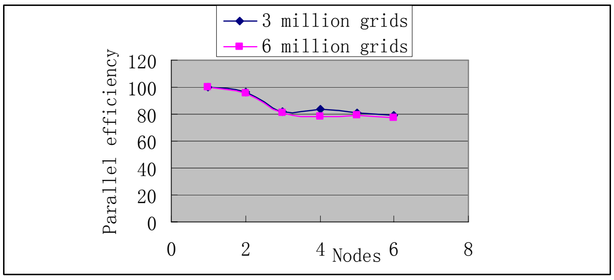

Figure6. Relationship between the number processor and parallel efficiency

The change of speedup in the experiment reveals that if the numbers of processors are fewer, the parallel computers will play better, whereas the processors increase, the performance of parallel machines declines gradually. Because the bus bandwidth of the shared memory machine is limited, if the CPU numbers are too many, so the memory access conflicts become intense, which has an impact on the speedup.

III. Conclusion

This study conducts the Contrast calculation by setting up a FDS parallel computing platform to prove the feasibility of building a FDS of parallel computing platform with high performance through lower configuration Computers. The conclusions drawn from the study are as following: (1) The parallel computing platform built up in this study has good expansibility, and the parallel efficiency has little change as the numbers of nodes (processors) increase; (2) The iteration time reduces significantly when the numbers of nodes are not very big (between 3 to 6 in numbers), and the parallel efficiency has little change as the scale of calculation (net grid number) varies; (3) When the number of CPU increases, the data exchange between the CPUs raises, and the traffic increases considerably, so the parallel efficiency reduces. Therefore, the calculation of a particular scale corresponds to a parallel node of optimal performance.

Acknowledgements

This work is supported by Guangdong Provincial Key Laboratory of Fire Science and Technology (2009A030302002, 2009A030302004, 2010A060801010).

References

[1] Pang Wen-jiang, Wu Jian-lin. An Analysis of the CFD Parallel Computing Platform and its Performance [J]. Journal of Chongqing University of Science and Technology (Natural Sciences Edition), 2009, 11(6):158-161.

[2] Maximilian Emans. Performance of parallel AMG-preconditioners in CFD-codes for weakly compressible flows [J]. Parallel Computing, 2010, 36(6):326-338.

[3] Nan Jiang. A Study on the Parallel Calculation Efficiency of Altix 50 in the Numerical Simulation for Three-dimensional Steady NS Laminar flow [J]. Aeronautical Computing Technique, 2010, 40(2):46-48.

[4] Zheng Qiu-Ya, Liu San-yang, Zuo Da-hai. CFD Parallel Computing and Load Balancing Research Using Multi-block Structured Grids [J]. Engineering Mathematics, 2010, 27(4):219-224.

[5] Yao Hao-wei, Liang Dong, Hu Zhi-jian. The Reconstruction of a Residential Building fire Based on Large Eddy Simulation[C]. International SympFsium on Fire Science and Fire Protection Engineering. 2009:257-261.

[6] K.McGrattan, B.Klein, J.Floyd, S.Hostikka. Fire Dynamics Simulator (Version 5)-User’s Guide. National Institute of Standards and technology. 2007.