The Performance of OpenFOAM in Beowulf Clusters

Ying Wei
Xiamen University Tan Kah Kee College
Xiamen, China
sailwy07@xujc.com

Feng Sha
Network Information Center
Xiamen University of Technology
Xiamen, China
shafeng@xmut.edu.cn

***Abstract*—With the acceleration of the process of industrial design, Computer-aided design software to be more widely used. At the same time, engineers thirst for speed is also growing strongly. In this article We analysis the OpenFOAM CFD software and detailed testing its performance on high-performance computers, the analysis results will provide a valid reference to the engineers, and there will be another choose of CFD software.**

Keywords-OpenFOAM; Parallel Computing; Performance

I. INTRODUCTION

After an investigation we found that The most widely use of computer-aided tools CFD production is Ansys, Fluent in all colleges and universities, research institutions, enterprises. The software is not only expensive, and its source code is not open. Therefore, out of strategic considerations, universities and other research institutions should pay more attention to the open source software to avoid commercial monopoly.

Industrial design at this stage is very complex，CFD software will always cost so many time in computing. Even in HPC ,it is common to run for several months. The running time is a primary consideration we must face. The performance of OpenFOAM in Beowulf Clusters is the focus of this paper.

II. THE FEATURE OF OPENFOAM

The OpenFOAM[1] (Open Field Operation and Manipulation) CFD Toolbox is a free, open source CFD software package produced by OpenCFD Ltd. It has a large user base across most areas of engineering and science, from both commercial and academic organizations. OpenFOAM has an extensive range of features to solve anything from complex fluid flows involving chemical reactions, turbulence and heat transfer, to solid dynamics and electromagnetic. It includes tools for meshing, notably snappyHexMesh, a parallelized meshed for complex CAD geometries, and for pre- and post-processing. Almost everything (including meshing, and pre- and post-processing) runs in parallel as standard, enabling users to take full advantage of computer hardware at their disposal.

III. BEOWULF CLUSTERS

A parallel system is built with six IBM HS22 blades. Each node has two Intel® Xeon™ Processor X5560 with 4-core CPU and 8GB PC3-10600 DDR3 memory. I/O system uses a HP DL380, which has dual processors (Intel® Xeon™ Processor X5650 with 6-core CPU) and 32GB memory. This node is also used as NFS server[2], NTP server, installation server and so on. All seven nodes are connected via two Gigabit Ethernet switches: IBM L2/3 Copper for IBM BCH and Nortel L2/3 Copper for IBM BCH. One Gigabit Ethernet for computation and the other is for communication [3].

In our system, all six IBM blade HS22 are used on computer nodes for taking advantage of the full computational capacity of Intel X5560 CPUs. On the other hand, the inherent disadvantage of the HP DL380 hardware for the use of I/O node influences the overall performance of the whole clusters. The performance degradation exhibits in two aspects: the bandwidth of the network, and the frequent hard disk reading/writing operations. To solve this problem, we installed and bound two Gigabit-network cards in HP DL380，and the time delay of the network is dependent on the high performance of the switch. Second, we built raid-0 on two SAS disks in HP DL380 to increase the speed of disk writing operation. The performance of the cluster can be close to 70% of the theoretical performance.

The operation system is Centos 5.4 x86_64. An xcat2 software was installed for IBM blade system hardware control, which is able to quickly operate hardware and install operation system and software. The Torque is used for job submission. Maui is used instead of Moab to manage the processes. The parallel file system is NFS. For inter-console message passing, we used the message passing interface (MPI) [4] version mpich2-1.2.1p1. We also installed mpiexec to start mpi without using mp. The compiler is GNU (gcc-4.4.3) and Inter ifortran (11.1.069) and intel mkl to increase Intel CPU performance [5].

IV.BUILT OPENFOAM IN BEOWULF CLUSTERS

The main tasks of this experiment are that. Firstly, Compile OpenFOAM in Beowulf Cluster. Secondly ,OpenFOAM should be managed by the PBS job scheduling system to complete the job submission, queuing, run, cancel, query and other operations. OpenFOAM can be run in parallel computing system. It is

Published by Atlantis Press, Paris, France.

relatively simple to install OpenFOAM on compute for run on one node, in OpenFOAM official website has more mainstream operating system to do the testing and optimization, like Ubuntu、SuSE、Fedora operating system. So we need to install OpenFOAM on Centos5.4 X86_64 by source code. In order to ensure the match and compatibility of the software, the OpenFOAM software itself is limited to several standard third-party software, including cmake、Gcc、gmp、paraview、Qt and so on. The version of these software is different in our Beowulf Clusters. Except mpich2, In order to ensure the software can reliably run the rest of the software we decide to use OpenFOAM default support third-party software. The Beowulf Clusters in our school is a multi-user parallel computer system . For running the OpenFOAM software users can change user environment variables file. If someone want to run other computing software is just to cancel a piece of code in the environment variable file. At last, We have completed the above two tasks. The software has run smoothly in our Beowulf Clusters.

The pbs script:

```
#!/bin/sh
#PBS -N openfoam
#PBS -o openfoam.log
#PBS -e openfoam.err
#PBS -q batch
#PBS -l nodes=4:ppn=1
#PBS -l walltime=2:00:00
#PBS -j oe
date
cd $WorkDir/0
mv alpha1.org alpha1
cd ..
blockMesh
setFields
decomposePar
mpirun   -np 4    interFoam -parallel    > run.log
reconstructPar
date
```

V. PARALLEL PERFORMANCE

In this study, we tested the performance of our IBM blade center clusters built by ourselves using fluid simulations of the system with the OpenFOAM/interFOAM package. The principle of the OpenFOAM parallel processing is the calculation of the regional block intend average assigned to run on the independent cores, the end of the calculation and then merge the data. To reduce disk operations across running across the computing nodes , each node can save the operating results on the node local disk until all nodes finished running and then calculate and combine the results.

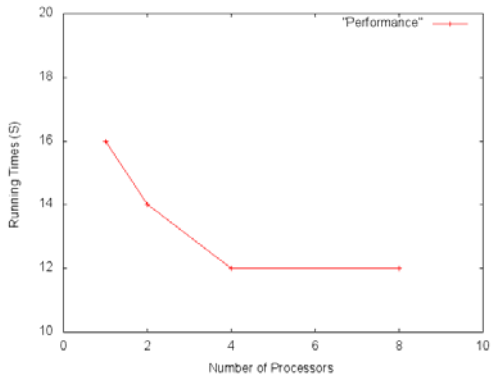

Figure1.The performance of OpenFOAM in parallel

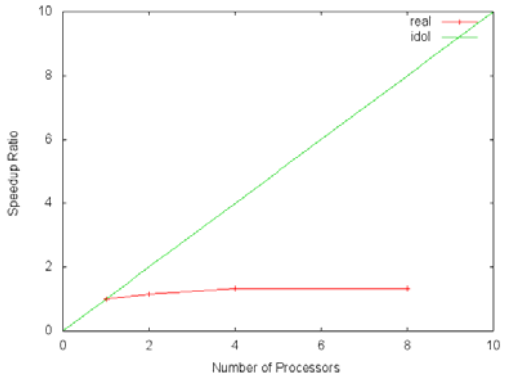

Figure2.The speedup of OpenFOAM in parallel

Figure 1 shows the running time as a function of the number of processors (ppn) varied from 1 to 8 for an interfoam run. As can be observed from Figure 1 when only one CPU of the node is used to run OpenFOAM program, it takes about 16 seconds for time. In this study, we applied message passing interface (MPI) and OpenFOAM on multi-processors on a single node. MPI assigns the computation duties on each CPU according to the parallel scheme designed in the interfoam package. For the same system, the computation time is highly shortened, while the speedup ratio in Figure 2 as well as the efficiency is enhanced. The computation time drops dramatically with the increase of processors to 2 and then starts to slowly decrease. With 8 processors, the computation approaches the plateau level, which demonstrates the number of processors needed for intefoam parallel computation.

VI. CONCLUSION

The test shows that OpenFOAM can run on high-performance computing platforms but accelerated is weak in our Beowulf Clusters than we thought. But OpenFOAM can be replace fluent in our platfoam. We currently has open the permision of using OpenFOAM in HPC of our collage for the teachers and students. More and more teachers and students will participate in this

study.

REFERENCES

[1] OpenFOAM [EB/OL] http://www.openfoam.com/.

[2] Calderon, A.; Garcia, F.; Carretero, J.; Perez, J. M.; Fernandez[J]. An Implementation of MPI-IO on Expand: A Parallel File System Based on NFS Servers . LECTURE NOTES IN COMPUTER SCIENCE . 0302-9743 ,20020101.

[3] Tao Wei, Parallel Molecular Dynamics Simulation of Lysozyme Hydration on IBM Blade Center Cluster[J],ICIS,2010.10.

[4] Rachel Gecker, Rethinking MPI [J],Corporate Meetings & Incentives , 0745-1636 ,2008.10.

[5] Ananth Grama. Introduction to Parallel Computing [M]. Machinery Industry Press,2005