FLUENT BENCHMARK ON AN AMD ATHLON64 SINGLE AND DUAL CORE WHITE BOX HPC CLUSTER

László Nagy[1], Péter Tóth[1], Máté Márton Lohász[2], Ákos Csécs[1]
[1]PhD student, [2]Assistant Professor
Budapest University of Technology and Economics, Mechanical Faculty, Department of Fluid Mechanics, Hungary

ABSTRACT

In this paper, we present a short benchmark performed on a 20 core white box cluster. The benchmark was carried out in the point of view of network usage, memory usage, hard disk usage during swapping in the case of parallel and single computing as well as the multi-core scalability to prescribe the directives for the further improvement. The base of this benchmark method is the commercially available ANSYS Fluent 6.3 computational fluid dynamics (CFD) solver test cases. It was found that dual core processor performs approximately 10% worse compared to single core ones.

1. INTRODUCTION

There were days when only the largest companies could afford supercomputers based on high-tech hardware, but in the last decade it has became possible to build a lower-cost high performance computing (HPC) cluster using x86 processors. The computer cluster is a group of linked computers sharing basic processing performance. The components of a cluster (nodes) are commonly connected to each other through fast local area network (Fast/Gigabit Ethernet, Myrinet and InfiniBand) led by a master node called front-end.

At the Department of Fluid Mechanics of the Budapest University of Technology and Economics a white box cluster has been developed in past 3 years to improve computing performance. The white box computer is a personal computer assembled from off-the-shelf parts [1]. The benefit of such a system is not only the performance compared to that provided by a single computer but its cost efficiency, because low-end video processors and common peripheries (keyboard and display) can be used. The cluster is mainly used for CFD applications, especially simulations using ANSYS Fluent. Fluent Inc. prepared a set of test cases ranging from small to more extensive simulation problems with different physics for benchmarking. The company publishes the results in a free database [2]. Our investigation is to use this benchmark tool to find the bottlenecks of the cluster performance and to show possible ways for its improvement.

2. FLUENT BENCHMARK

The Fluent Benchmark package can be used to analyze the applied computer system, in this case our HPC cluster system. Only two different cases were chosen from the available nine to perform the in-house benchmark (Table 1).

Table 1. The applied cases from Fluent Benchmark

Benchmark	Cells	Mesh	Models	Solver	Description
FL5M2	242.782	Hybrid, hanging node	k-ε	Segregated, implicit	Turbulent flow in an engine valve port
FL5L2	3.618.080	Hybrid	RNG k-ε	Segregated, implicit	External aerodynamics around a car body

Many parameters exist in information technology to measure the processor speed. The speed is commonly given in thousand instructions per second (kIPS), million instructions per second (MIPS). But in the fields of scientific calculations the FLOPS (FLoating point Operations Per Second) is used [3]. As an example an Athlon 64 X2 3800+@2.2GHz processor is characterized by 18.900 MIPS. The *Rating*, defined in Eq.1 is the primary metric which will be used to characterize the performance of the Fluent benchmark cases. The rating shows the number of test cases that could be run in a day.

$$Rating = \frac{86400}{Iter * M}, \tag{1}$$

where *Iter* is the number of seconds required for one iteration in the benchmark and *M* is the number of the iterations, which is actually 25 in every benchmark case. The *Speedup* is the ratio of the wall-clock time required to complete a given calculation using a single core compared to that of the equivalent calculation performed on a concurrent machine (Eq.2.). The rating and speedup are linear:

$$Speedup = \frac{Rating_N}{Rating_1}. \tag{2}$$

The efficiency is characterized by the parallelism of the simulation (Eq.3)

$$\eta = \frac{iter_1}{iter_N N}, \tag{3}$$

where $iter_N$ is the iteration time for running the case on *x* processor cores, *N* is the number of applied processor cores. In our benchmark we followed these Fluent Benchmark notations for comparison with the database [2].

The benchmark package can be installed on any system with a functionally working Fluent environment. All of the test cases and the scripts are included in the package. The script runs the given benchmark case for any prescribed parallel core numbers and collects all of the results in a file. The file starts with a header containing the main parameters of the case (code version, size, number of cores and the summary of the rating), and the body of the file is a table of different performance indicators in the function of core numbers.

3. CLUSTER HARDWARE AND SOFTWARE

As our previous experience and the literature shows [4], AMD processors perform better in the case of parallel computations. This is the reason that the AMD platform was chosen to build up our HPC cluster.

Each computer has a single processor and two different types of processors are used in the two different racks of the cluster. Dual core processors are denoted by *compute-0-* (_c0_)* and single core processors by *compute-1-* (_c1_)* (Table 2). Both kinds of processors have 512KB level-1 cache for each core. All of the computers have 8GB swap memory.

Table 2. Hardware configuration

Name	CPU	Core(s)	#	Clock	Memory
Front-end	AMD Athlon 64 3000+	1	1	2250MHz	2GB
Compute-0-* (_c0_)	AMD Athlon 64 X2 3800+	2	5	2000MHz	3GB
Compute-1-* (_c1_)	AMD Athlon 64 3000+	1	10	2250MHz	2GB

The interconnection between the computer nodes and the front-end are realized using an Ethernet (on CAT6 cable) connection via a 3COM Gigabit Baseline switch (version 3C16479) with 24 equivalent gigabit ports [5]. Direct CPU connected motherboard integrated Ethernet cards are used in each computer.

The applied Linux distribution is the CentOS based ROCKS 4.2.1 open source cluster operation system using 2.6.9 kernel. The Sun Grid Engine (SGE) [6] is used as a queuing system responsible for distributing the submitted jobs between the cores of the computers.

4. RESULTS

During the benchmark we had to control both the network and I/O traffic on the cluster system. One user was dedicated to run the benchmark cases without any rivalry simulations. As was mentioned in the last section, two different test cases were used for benchmarking.

The FL5M2 test case was run three times from 1 to 10 cores chosen from the *c1 rack* (test#2, see Table 3) for checking the repeatability of the results. The uncertainty is increased with the number of cores and the maximum deviation from average of the three cases is 2.3%. This issue shows that the benchmarks are representative using a single run.

In the lack of any network monitor indirect network analyses (test#3) were done. A competition for the network traffic was prepared: the same FL5M2 test case was run for two times 5 cores in cabinet 1, producing an average rating of 3055. This was compared to the stand-alone 5 cores test with a rating of 3141 (average). This low difference (2.8%) indicates the negligible influence of network traffic. There are two competing effects in dual core processor simulations. The first one is that cores inside a processor can communicate faster than cores between different processors (this is an advantage). This was measured by running a two

partitions case on 1 or 2 processors (test#4). The simulation running on one processor was 1.8% faster, meaning that the importance of this effect is low.
The other effect is that the two cores of the processors use the same network card for communication between other cores. This effect was investigated by running two rival cases on the same processors (test#4) using 5-5 cores from *c0 rack*. The case where two identical simulations are competing for the network card had an average rating of 1566 compared to the same case without competition, with a rating of 2583. This 40% reduction is significant.

Table 3. The rating of the benchmarks (1-10 cores)

Test Nr.	Benchmark name	Rating in the function of number of cores										
		serial	1	2	3	4	5	6	7	8	9	10
#1	fl5l2_c_all			150	213	290	360	368	419	493	500	475
#1	fl5l2_c1			239	330	405	479	549	549	638	623	604
#6	fl5l2_c1_2c_swapno			**171**								
#6	fl5l2_c1_2c_swapyes			**66**								
#1	fl5m2_c_all											
#4,5	fl5m2_c0_1c	818	836	1155	1816	2286	**2583**					
#4	fl5m2_c0_1c_2job_5CPU_1						**1545**					
#4	fl5m2_c0_1c_2job_5CPU_2						**1587**					
#5	fl5m2_c0_2c	819	836	1176	1859	2228	2792	2624	3608	4033	3954	4403
#7	fl5m2_c0_c1_mix2			**1205**								
#7	fl5m2_c0_c1_mix3				**2052**							
#7	fl5m2_c0_c1_mix4					**1955**						
#7	fl5m2_c0_c1_mix8									**4403**		
#3	fl5m2_c1_1c_2job_5CPU_1						**3150**					
#3	fl5m2_c1_1c_2job_5CPU_2						**2959**					
#1,2	fl5m2_c1_run1	901	923	1289	2057	2743	**3130**	3716	4272	5023	5468	**5868**
#2	fl5m2_c1_run2	901	921	1282	2124	2734	**3148**	3716	4283	5075	5539	**5694**
#2	fl5m2_c1_run3	901	923	1289	2100	2679	**3145**	3696	4299	5060	5400	**5918**

In practical situations these effects play a simultaneous role. An important question is how to dedicate cores to a job. One environmental variable of SGE controls the process. The *filled* option of the variable means that the SGE tries to fill up the entire free core in each distributed multi core processors. *Round robin* variable setting means that first all processors get a job on at least one core before a second job is directed to any processor. In test#5 these two settings are compared. In this test both of the two above mentioned effects are present. The result is visualized in Fig. 3, where the "*_1cores*" and "*_2cores*" columns have to be considered, where "*1_cores*" in accordance with using *round robin* SGE option and "*_2cores*" in accordance with using *fill* option. For the 3 and the 5 core cases the "_2cores" version is faster but for 4 cores the opposite is true. This highlights that the competition of the two effects is not simple to predict. Further information could be gained by running the same tests in a rivalry manner.

In the next test (#1) the dual core processors (*c0 rack*) were compared to the single core ones (*c1 rack*), the dual core processors being used in the "_2cores" manner. The result is depicted in Figs 1-3. It is clear from both cases (FL5M2 and

FL5L2) that for core numbers higher than 5 the speedup for dual core processors is significantly (~10%) worse than for the single one. A possible explanation for this is that the reduced network card availability for one core effect is more important than the gain by the other effect; this is in agreement with results of the basic tests (#4). Considering Fig. 3, the parallel efficiency [7,8] on its own can be judged as well, and shows a dramatic reduction from 2 cores.

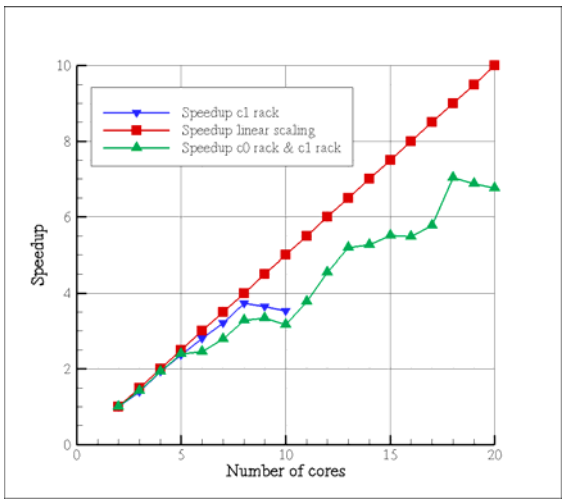

Fig. 1 Scalability for FL5L2 test case

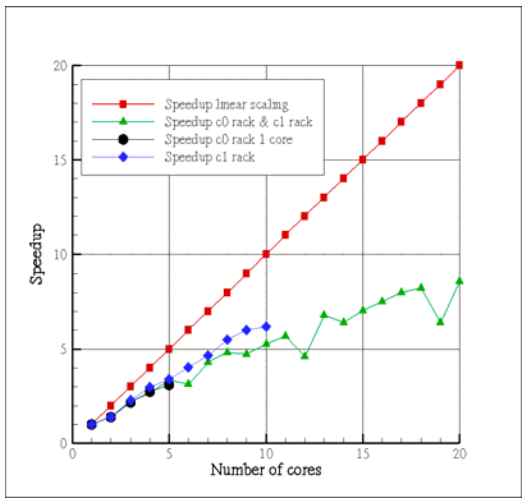

Fig. 2 Scalability for FL5M2 test case

In the daily routine usage of the cluster, the typical situation is that processor cores of mixed types are assigned to a typical job. To investigate this situation test#7 was prepared, and its results are depicted in Fig. 4. This figure enables the comparison of the mixed to the homogeneous cores. It has to be remarked that the speed of the two kinds of processors is different; it was expected that the combined performance would be between the two homogenous processor results. The only deviation from this was when running on 4 cores; a possible explanation could be in the partitioning of the mesh file.

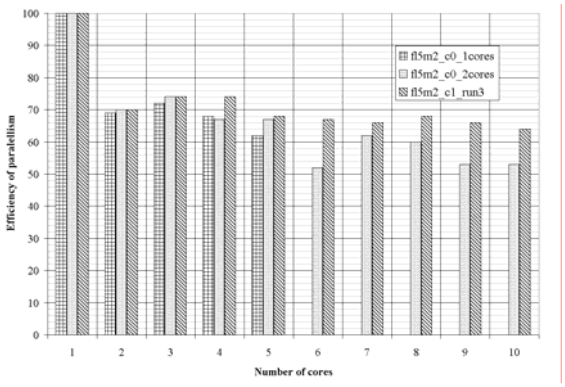

Fig. 3 Efficiency of parallelism

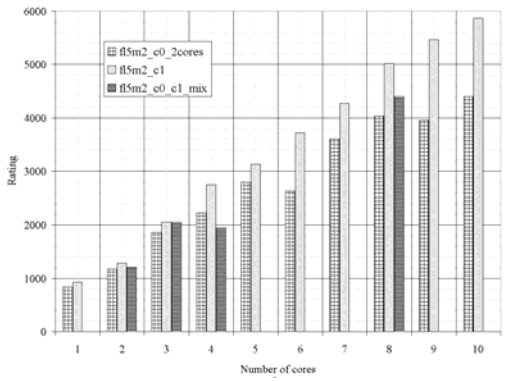

Fig. 4 Hybrid configuration

One test investigated the effect of swapping (test#6). Two computers were chosen to test usage of swapping from compute-1-* cabinet. To force the swapping the physical memory was limited to 1GB instead of 2GB (by physically removing the memories from the motherboard). The rating of the FL5L2 test case was 66 in

the case of the hard swapping, against 171 points when the 2GB RAM was used (without swapping). This obviously means that hard swapping must be avoided.

5. CONCLUSIONS AND RECOMMENDATIONS FOR IMPROVEMENT

In the present paper the parallel performance of AMD dual and single core processors was compared in CFD applications. It was found that dual core processors are worse due to the reduced available network resource to each core. However, computers built from multi core processors are notably cheaper. A possible remedy to compensate this deficiency would be to use different network card solutions (more or faster cards on a motherboard). Another possible way could be to use multi socket motherboards, which could eliminate the from-motherboard communications for small cases (all the cores would be on the same motherboard).
Such a database can be used not only to prepare further hardware improvement, but the queuing system can be also configured optimally with knowledge of the system performance.

It becomes also obvious that swapping needs to be completely avoided. The two possibilities are: to extend the physical memory assigned to each core or to distribute the job to more cores.

ACKNOWLEDGEMENTS

We wish to thank Csaba Erdei for helping us to build up the background of the Linux system. Financial support from the CFD.HU LTD and the Hungarian National Fund for Science and Research under contract No. OTKA T 30075 to build our cluster system is also acknowledged.

REFERENCES

[1] DENG Y., KOROBKA A.: **The performance of a supercomputer built with commodity components.** Parallel Computing 27, 2001, pp.91-108.

[2] **Fluent Benchmark database.** www.fluent.com/software/fluent/fl5bench/ (10 January 2008)

[3] FODOR Z., KATZ S.D., PAPP G.: **Better than $1/Mflops sustained: ascalable PC-based parallel computer for lattice QCD.** Computer Physics Communications 152, 2003, pp.121-134.

[4] EHRIG M.: **CFX Performance on Multiple Platforms Comparing different Architectures.** ANSYS Conference & 25. CADFEM User's meeting, 21-23. November 2007. Dresden, Germany, ISBN 3-937523-04-9

[5] FATOOHI R., KARDYS K., KOSHY S., S. SIVARAMAKRISHNAN S., VETTER J.S.: **Performance evaluation of high-speed interconnects using dense communication pattern.** Parallel Computing 32, 2006. pp.794-807.

[6] **Sun Grid Engine**. www.sun.com/software/gridware/ (10 January 2008)

[7] BRIGHTWELL R., PLIMPTON S.: **Scalability and Performance of two large Linux cluster.** Journal of Parallel Distribution Computing 61, 2001, pp.1546-1569.

[8] **Cluster computing for CFD HP reference.** http://www.fluent.com/software/fluent/fl5bench/otherart/hpclust.pdf (10 January 2008)