Acta
Technica
Jaurinensis

Vol. 9, No. 1, pp. 42-53, 2016
DOI: 10.14513/actatechjaur.v9.n1.394
Available online at acta.sze.hu

Measuring the Efficiency of Parallel Discrete Event Simulation in Heterogeneous Execution Environments

G. Lencse[1], I. Derka[1]

[1]Széchenyi István University, Department of Telecommunications
Egyetem tér 1., 9026 Győr, Hungary
Phone: +36 96 503 400
e-mail: steve@sze.hu

Abstract This paper deals with the experimental testing and verification of the earlier proposed load balancing and coupling factor criteria for the conservative parallel discrete event simulation in heterogeneous execution environments whether they can ensure a good speedup. The definition of the relative speedup is extended to the heterogeneous systems in a natural way. This definition is used to measure the efficiency of the parallel simulation executed by heterogeneous systems. A closed queueing network is used as the simulation model. It is demonstrated on the example of a heterogeneous system containing 134 CPU cores of 6 different types of computers that a good speedup can be achieved using the load balancing and coupling factor criteria. It is shown that the extension of the definition of the relative speedup to the heterogeneous systems made it easy to judge the speedup of parallel discrete event simulation in heterogeneous execution environments.

Keywords: parallel discrete event simulation, heterogeneous execution environment, conservative synchronisation method, load balancing criterion, coupling factor criterion

1. Introduction

Event-driven discrete event simulation (DES) is a powerful method for the performance analysis of information and communication technology (ICT) systems. The detailed modelling and simulation of these systems often requires a huge amount of computing power and memory. Parallelisation can be a natural solution. Kunz [1] points out that as the ongoing development in the hardware sector favours an increasing number of processing units over an increasing speed of a single unit thus the parallel simulation will remain an important and active field of research.

However, because of the algorithm of the event-driven DES, parallel discrete event simulation (PDES) it is not an easy task and the achievable speedup is often limited. When doing PDES, the model of the system is divided into partitions (called Logical Processes), and the partitions are assigned to processors that are executing them. To maintain causality, the virtual times of the partitions must be synchronised. There are different methods for synchronisation [1]. The conservative method ensures that causality is never violated. An event can be executed only if we are certain that no events with smaller timestamp exist (and also will not be generated) anywhere in the model. Unless the simulated system has a special property that the so called lookahead is large enough, the processors executing the partitions need to wait for each other in the majority of time, so the achievable speedup is poor.

A method for assessing available parallelism in a simulation model for conservative synchronization was proposed in [2]. The method requires only a small number of parameters that can be measured on a sequential simulation easily. The results of the aforementioned work were checked for homogeneous clusters up to 24 CPU cores in [3] and it was also examined how the different parameters of the model influence the achievable speedup. Criteria for a good speedup in a heterogeneous execution environment were proposed in [4]. The criteria were justified by several measurements in a test system. Further results on the examinations of different factors that influence the achievable speedup of parallel discrete event simulation in a heterogeneous execution environment were presented in [5]. Moreover, the definition of the relative speedup was extended to heterogeneous systems and this extension was used in the discussion of the results the experiments to evaluate the efficiency of parallel simulation executed by heterogeneous systems in the same conference paper. In our current paper, conference paper [5] is extended with experiments using significantly higher number of CPU cores (however the different kinds of experiments published in that paper are not included in this paper due to space limitations).

The remainder of this paper is organised as follows: first, a brief summary of the method for assessing the available parallelism is given. Second, our concept of heterogeneous execution environment, our criteria for a good speed up and our previous results are summarized. Third, the definition of the relative speedup is extended to the heterogeneous systems to be able to express the efficiency of parallel simulation executed by heterogeneous systems. Fourth, our heterogeneous test environment and simulation model are described. Fifth our measurements taken on the largest heterogeneous cluster are presented and discussed. Sixth, super-linear speedup is demonstrated on a large homogeneous cluster. Finally, our paper is concluded.

This topic was identified as being of importance in the parallel simulation of large systems using heterogeneous execution environments.

2. Method for Assessing Available Parallelism

The available parallelism can be assessed using some quantities that can be measured during a sequential simulation of the model in question. The following description is taken from [3].

The paper [2] uses the notations *ev* for events, *sec* for real world time in seconds and *simsec* for simulated time (model time) in seconds. The paper uses the following quantities for the assessing of available parallelism:

- P performance represents the number of events processed per second (*ev/sec*).
- E event density is the number of events that occur per simulated second (*ev/simsec*).
- L lookahead is measured in simulated seconds (*simsec*).
- τ latency (*sec*) is the latency of sending a message from one Logical Process (LP) to another.
- λ coupling factor can be calculated as the ratio of LE and τP:

$$\lambda = \frac{L \cdot E}{\tau \cdot P} \tag{1}$$

It was shown in [3] that if λ is in the order of several hundreds or higher then we may expect a good speedup. It may be nearly linear even for higher number of segments (N) if λ_N is also at least in the order of several hundreds, where:

$$\lambda_N = \frac{\lambda}{N} \tag{2}$$

3. Modelling and Simulation in Heterogeneous Execution Environments

This chapter is a summary of [4].

3.1. Our Concept of Heterogeneous Execution Environments

A logical topology of two levels was proposed: a *star shaped network* of *homogeneous clusters*. This model is simple enough and can describe a typical *heterogeneous execution environment*. What is logically described as a homogeneous cluster, it can be physically, for example, a cluster of PCs with identical configuration interconnected by a switch or it can be a chassis based computer built up by several main boards, etc. The main point is that a homogeneous cluster is built up by identical configuration elements especially concerning CPU type and speed as well as memory size and speed. The homogeneous clusters are interconnected logically in a star shaped topology. The physical connection can be a switch or the topology may be different but our model considers it to be a star for simplicity.

3.2. Criteria for Achieving a Good Speedup

Two criteria were set up. The ***load balancing criterion*** requires that all the CPUs (or CPU cores) should get a fair share from the execution of the simulation. A fair share is proportional to the computing power of the CPU *concerning the execution of the given simulation model*. (This is very important, because, for example, using different

benchmark programs for the same set of computers one can get seriously different performance results.) Thus, for the fair division of a given simulation model among the CPUs, the CPUs should be benchmarked by the same type of simulation model that is to be executed by them (but smaller in size, of course). The *lookahead* or ***coupling factor criterion*** is the same as presented and tested in [3] up to 24 CPU cores.

3.3. Most Important Results

The load balancing criterion was justified by measuring the execution time of a model with different partitioning. The results of our experiments were quite close to the values computed according to the load balancing criterion. The coupling factor criterion was justified by different scenarios including a simulation executed by 64 CPU cores of 4 types resulting in a good speedup in [4].

4. Efficiency of Parallel Simulation Executed by Heterogeneous Systems

4.1. Relative Speedup of Program Execution by Heterogeneous Systems

First, the definition of the relative speedup of parallel execution of programs is extended for heterogeneous systems (in general, not only for simulation).

The conventional definition of the *speedup* (s_n) of parallel execution is the ratio of the speed of the parallel execution by n CPUs and the sequential execution by 1 CPU that is equal with the ratio of the execution time of the sequential execution (T_1) and that of the parallel execution (T_n):

$$s_n = \frac{T_1}{T_n} \tag{3}$$

The *relative speedup* (r_n) can be calculated as the ratio of the speedup and the number of the CPUs that produced the given speedup:

$$r_n = \frac{s_n}{n} \tag{4}$$

The relative speedup measures the *efficiency* of parallel execution. A relative speedup value of 1 means that the speedup is linear that is the computing power of n CPUs can be fully utilized.

When dealing with heterogeneous systems, not only the number of the CPUs but also their performance is to be taken into consideration. We were looking for a definition of the relative speedup of heterogeneous systems that can be used to measure the efficiency of program execution by the heterogeneous systems in the same way: its value of one should mean that the computing power of all the CPUs (from different types) can be fully utilized.

Let us denote the *number of the CPU types* in a heterogeneous system by *NT*, the *number* and the *performance* of CPUs available from *type i* by N_i and P_i, respectively. The *cumulative performance* of the heterogeneous system is:

$$P_c = \sum_{j=1}^{NT} P_i \cdot N_i \tag{5}$$

Let us denote the *execution time* of a given program by a single CPU of *type i* by T_i and let T_h denote the *execution time* of the given program by the *heterogeneous system*. The speedup of the heterogeneous system compared to the sequential execution by one CPU of type *i* is:

$$s_{h/i} = \frac{T_i}{T_h} \tag{6}$$

The relative speedup of the heterogeneous system against the sequential execution by one CPU of type *i* is now defined as:

$$r_{h/i} = \frac{T_i \cdot P_i}{T_h \cdot P_c} \tag{7}$$

We believe that this definition can be used in general for measuring the efficiency of program execution by heterogeneous systems. Thus, for simplicity, we used the word "CPU" in this section, but the expression "CPU core" could be used instead, as it is used everywhere else in this paper.

4.2. Measuring the Efficiency of Parallel Simulation Executed by Heterogeneous Systems

If the above definition of the relative speedup of program execution by heterogeneous systems is used for measuring the efficiency of parallel simulation executed by heterogeneous systems and the performance values of the CPU cores from different types are measured by benchmarking them with the same simulation model (expressing its value in events per seconds and the value of execution time in seconds) then the numerator of expression (7) gives the same values for all the values of *i* (that is its value is independent of the CPU core types): it is equal with the total number of events in the sequential simulation.

Note that the number of events in the parallel version of the simulation may be higher (e.g. due to communication and synchronization overhead) but only the events of the original sequential simulation are the essential part of the operation of the original model. Thus efficiency should consider the events of the original sequential simulation only.

Denoting the total *number of events* in the sequential simulation by N_E, definition (7) can be rewritten as:

$$r_h = \frac{N_E}{T_h \cdot P_c} \tag{8}$$

Thus, we have shown that the value of relative speedup calculated by (8) does not depend on which CPU core it was calculated against, it characterises the parallel simulation itself. Note that this is still true if one uses definition (7), thus one can use any of them selecting on the basis of which values are easier to measure directly in the given simulation.

5. Heterogeneous Test Environment

5.1. Available Hardware Base

The following servers, workstations and PCs were available for our experiments. Note that this hardware base is significantly larger than that of [5].

One Sun Server SunFire X4150 (referred as: BigSun)

- Two Quad Core Intel Xeon 2.83GHz CPUs
- 8GB DDR2 800MHz RAM
- Two near-line SAS 160GB HDD
- Two Intel 82571EB Gigabit Ethernet NIC
- Two Intel 80003ES2LAN Gigabit Ethernet NIC

Altogether it means a homogeneous cluster of 8 cores.

Thirteen LS20 Blades in an IBM BladeCenter (referred as: IBM)

- Two Dual Core Opteron 280 2.4GHz CPUs
- 4GB DDR2 667MHz RAM
- 73GB SCSI Ultra 320 HDD
- Broadcom NetXtreme BCM5704S Gigabit Ethernet NIC

Altogether it means a homogeneous cluster of 52 cores.

Ten Dell Precision 490 Workstations (referred as: Dell)

- Two Intel Xeon 5140 Dual Core 2.33GHz CPUs
- 4x1GB DDR2 533MHz RAM (quad channel)
- 80GB SATA HDD
- Four Intel 82571EB Gigabit Ethernet NICs
- Broadcom NetXtreme BCM5752 Gigabit Ethernet NIC

Altogether it means a homogeneous cluster of 40 cores.

Eleven AMD PCs (referred as: AMD)

- AMD Athlon 64 X2 Dual Core 4200+ 2.2GHz CPU
- 2GB DDR2 667 MHz RAM
- Two 320 GB SATA HDD
- nVidia CK804 Gigabit Ethernet NIC

Altogether it means a homogeneous cluster of 22 cores.

One Dell Precision 490 Workstation with faster CPUs (referred as: FastDell)

- Two Intel Xeon 5160 Dual Core 3GHz CPUs
- 4x1GB DDR2 533MHz RAM (quad channel)
- 80GB SATA HDD
- Four Intel 82571EB Gigabit Ethernet NICs
- Broadcom NetXtreme BCM5752 Gigabit Ethernet NIC

Altogether it means a homogeneous cluster of 4 cores.

Two Sun Servers SunFire X4200 (referred as: LittleSun)

- Two 2.2GHz Dual Core AMD Opteron 275 CPUs
- 4x1GB 400MHz ECC SDRAM
- 80GB HDD
- Four Intel 82546EB Gigabit Ethernet NICs

Altogether it means a homogeneous cluster of 8 cores.

Switches for interconnection:

- Two 3Com Baseline 2948 SFP Plus 48 ports Switches (3CBLSG48)
- Cisco Intelligent Gigabit Ethernet Switch Module, 4 ports (Part Number 32R1894) in the BladeCenter

5.2. Software Environment

Operating Systems

Debian Squezee (x86_64) Linux was used on all the computers.

Cluster Software

OpenMPI 1.6.2 (x86_64)

Discrete-Event Simulation Software

The widely used, open source OMNeT++ 4.2.2 discrete-event simulation environment [6] was chosen. It supports the conservative synchronization method (the Null Message Algorithm) since 2003 [7] We also expect that because of the modularity, extensibility and clean internal architecture of the parallel simulation subsystem, the OMNeT++ framework has the potential to become a preferred platform for PDES research.

5.3. The Simulation Model

Bagrodia and Takai proposed the Closed Queueing Network for testing the performance of conservative parallel discrete-event simulation [8]. OMNeT++ has a CQN implementation among its simulation sample programs. We have found this model perfect for our purposes and it was used in our current paper as well as in [3]-[5]. The below description of the model is taken from [3].

This model consists of *M* tandem queues where each tandem consists of a switch and *k* single-server queues with exponential service times (Fig. 1, left). The last queues are looped back to their switches. Each switch randomly chooses the first queue of one of the tandems as destination, using uniform distribution. The queues and switches are connected with links that have nonzero propagation delays. The OMNeT++ model for CQN wraps tandems into compound modules.

To run the model in parallel, we assign tandems to different LPs (Fig. 1, right). Lookahead is provided by delays on the marked links.

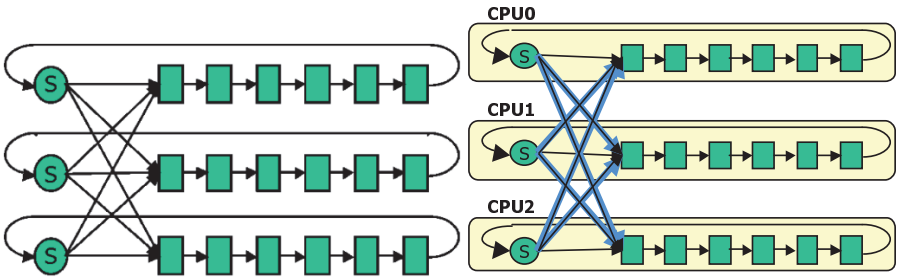

Figure 1. M=3 tandem queues with k=6 single server queues in each tandem queue – and its partitioning (taken from [3])

As for the parameters of the model, the preset values shipped with the model were used unless it is stated otherwise. Configuration B was chosen, the one that promised good speedup.

6. Experiments and Results of our Largest Heterogeneous Cluster

Fig. 2 shows the interconnection of the elements of our heterogeneous environment.

The automatic CPU frequency scaling [9] was switched off on all the computers by setting the CPU clock frequency to its maximum value. Unlike before, now Debian Squeeze (x86_64) operating system was installed on all the computers.

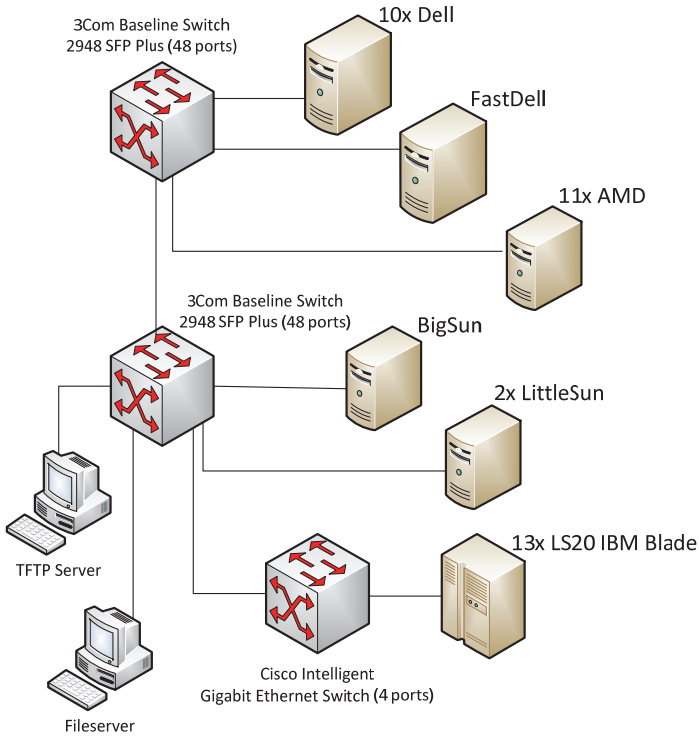

Figure 2. The elements of the heterogeneous execution environment with their interconnections

The number of the single queues in the tandem queues was also revised compared to papers [4] and [5] for the following reasons. In paper [3], M=24 tandem queues were used having each k=50 queues in them. Since then, the number of tandem queues were increased up to 480 (so that they can be distributed among the high number of cores) but the number the single queues were kept. This resulted in a significant change in the ratio of the two values. Now, the original ratio was restored by using the following values: M=480, k=1000. L=2000s was kept from the last test in [5]. As the expected value of the service time of the single queues was kept 10 seconds, the effect of the vacationing jobs (see [3]) was significantly reduced, because now the events could spend much more time in the tandem queues then in the long delay lines.

All the CPUs were benchmarked using a smaller network with the following parameters: M=24, k=1000, L=2000*sec* and the test were executed until 10^4*simsec*. All the tests were executed 11 times and average and standard deviation were calculated. The results are shown in Table 1. These performance values were used for the performance proportional partitioning of the simulation model containing 480 tandem queues among the 134 cores of our largest heterogeneous cluster. The partitioning is shown in Table 2.

Table 1. The Benchmarking Results (ev/sec) for the Largest Heterogeneous Cluster

Core Type	***BigSun***	***IBM***	***Dell***	***AMD***	***FastDell***	***LittleSun***
Average	264 687	176 118	205 116	173 764	240 491	173 540
Std. Dev.	1 762	1 221	1 631	1 686	1 472	1 768

Table 2. The Division of the 480 Tandem Queues Among the Cores of the Largest Heterogeneous Cluster

Core type	***P_i (ev/sec)***	***N_i***	***n_i***	***no. of cores***	***tandems /core***	***cumulated tandems***
BigSun	264 687	8	4.95	8	5	40
IBM	176 118	52	3.30	38	3	114
				14	4	56
Dell	205 116	40	3.84	40	4	160
AMD	173 764	22	3.25	22	3	66
FastDell	240 491	4	4.50	4	5	20
LittleSun	173 540	8	3.25	8	3	24
No. of all the cores:		134	Number of all the tandems:			480

In order to keep the total sum of number of the tandem queues exactly 480, we could not always follow the rules of the mathematical rounding: 4 tandem queues were assigned to each one of 14 CPU cores from among the 52 cores of the IBM blades whereas only 3 tandem queues were assigned to each one of the other 38 CPU cores from the same type (which one matches with the result of the correct rounding of 3.3).

Using the 134 cores, 10^5*simsec* long simulations of the CQN model were executed eleven times and average and standard deviation of the execution time were calculated. The average execution time was 278.19*sec* with the standard deviation of 2.63*sec* (which is less than 1% of the average).

For the calculation of the relative speedup we need either N_E the total number of events in the sequential execution of the simulation or the execution time of the sequential simulation using one of the CPU cores. We executed the sequential simulation of the same model and number of events was: N_E=6.1675*10^9*ev*. Using the cumulative sum of the performance of 134 cores, P_c=25.653*10^6*ev/sec*, the relative speedup of the heterogeneous cluster is:

$$r_h = \frac{6.1675 \cdot 10^9 \, ev}{278.19 \, sec \cdot 25.653 \cdot 10^6 \, ev / sec} \approx 0.8642 \tag{9}$$

This is an excellent result produced with 134 CPU cores of 6 different types of computers. For comparison, the relative speedup of the heterogeneous cluster was 0.88 using only 86 CPU cores from 4 types in [5].

7. Testing a Large Homogeneous Cluster

Finally, we tested a large homogeneous cluster. Even though we had 13 blades, we used only 12 of them (having 48 cores) to be able to evenly divide the 480 tandem queues. The parameters of the model were kept from the previous experiment. The average execution time of the 11 simulation runs was 730.68*sec* with 3.22*sec* standard deviation (which is 0.44% of the average) whereas the execution time of the sequential simulation performed by a single IBM Blade core was 52077*sec* (!). This means a 71.27 speedup produced by a system having only 48 core with is about 1.5 relative speedup. This phenomenon of the super linear speedup can be easily explained by the effect of caching: the working set of the simulation program can better fit into the cache of the 48 CPU cores than into that of a single CPU core. Similar phenomenon was reported in [10], see page 95.

8. Conclusion

An extension of the relative speedup for heterogeneous systems was introduced. It proved to be an appropriate tool for the evaluation of the speedup of parallel discrete event simulation in heterogeneous execution environments. We could achieve 0.86 relative speedup using a heterogeneous cluster built up by 134 CPU cores of 6 different types of computers. We also demonstrated a super-linear speedup (relative speedup of 1.5) on a 48 core homogeneous cluster.

References

[1] Kunz G: Parallel discrete event simulation, in K. Wehrle, M. Günes and J. Gross (Eds.) Modeling and Tools for Network Simulation, Berlin, Springer-Verlag, pp. 121-131, 2000

[2] Varga A, Sekercioglu YA, Egan GK: A practical efficiency criterion for the null message algorithm, in Proc. 15th European Simulation Symposium, Delft, Netherlands, pp. 81-92, 2003

[3] Lencse G, Varga A: Performance prediction of conservative parallel discrete event simulation, in Proc. 2010 Industrial Simulation Conference, Budapest, Hungary, pp. 214-219, 2010

[4] Lencse G, Derka I, Muka L: Towards the efficient simulation of telecommunication systems in heterogeneous execution environments, in Proc. 36th International Conference on Telecommunications and Signal Processing, Rome, Italy, pp. 304-310, 2013

[5] Lencse G, Derka I: Testing the speedup of parallel discrete event simulation in heterogeneous execution environments, Proc. 11th Annual Industrial Simulation Conference, Ghent, Belgium, pp. 101-107, 2013
[6] Varga A, Hornig R: An overview of the OMNeT++ simulation environment, Proc. 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops, Marseille, France, 2008, http://dl.acm.org/citation.cfm?id=1416290
[7] Sekercioglu YA, Varga A, Egan GK: Parallel simulation made easy with OMNeT++, Proc. 15th European Simulation Symposium, Delft, The Netherlands, pp. 493-499, 2003
[8] Bagrodia RL, Takai M: Performance evaluation of conservative algorithms in parallel simulation languages, IEEE Transactions on Parallel and Distributed Systems, Vol. 11, No. 4, pp. 395-411, 2000
[9] Archlinux, CPU frequency scaling, https://wiki.archlinux.org/index.php/CPU_frequency_scaling
[10] Benzi J, Damodaran M: Parallel three dimensional direct simulation Monte Carlo for simulating micro flows, in Parallel Computational Fluid Dynamics 2007, Springer, 91-98, 2009
DOI: 10.1007/978-3-540-92744-0_11