DYNAMIC CONFIGURATION IN A LARGE SCALE DISTRIBUTED SIMULATION FOR MANUFACTURING SYSTEMS

Koichi Furusawa*
Kazushi Ohashi†
Mitsubishi Electric Corp. Advanced Technology R&D Center
8-1-1, Tsukaguchi-honmachi
Amagasaki, Hyogo 661-8661, Japan
E-mail: *Furusawa.Koichi@wrc.melco.co.jp
† Ohashi.Kazushi@wrc.melco.co.jp

KEYWORDS
Dynamic configuration, Distributed simulation, Integrated simulation, Manufacturing, Synchronization

ABSTRACT

We are developing an integrated simulation environment for manufacturing systems, in which simulators are synchronized in order to guarantee timed consistency among the simulators. We propose a dynamic configuration method in distributed simulation. It automatically configures distributed simulators during integrated simulation and enables efficient execution of large scale integrated simulation. In this paper, we illustrate the proposed dynamic configuration method, and we show its evaluation results under various conditions.

INTRODUCTION

Manufacturing systems have become increasingly large and complicated, and they consist of various kinds of equipment. We usually use simulation technology to verify their behavior when setting up a new factory or carrying out improvements. High accuracy simulation is necessary to reduce implementation costs.

Many simulators have been already developed in the manufacturing field. They are not easy to combine with each other for an integrated simulation because of the asynchronous execution of the simulators. To solve the problem, several methods of synchronization between simulators are proposed (Carothers et al. 1997, Dahmann et al. 1997, Defense Modeling and Simulation Office 1998). We proposed more efficient synchronization mechanism for manufacturing systems (Furusawa and Yoshikawa 2002).

In simulation for manufacturing systems, various kinds of simulators are integrated. The integrated simulators are so many that they are necessary to be distributed to several PCs in order to reduce the simulation load. The configuration of the distributed simulators are usually decided by the user of the simulators. It is not, however, easy to estimate the PC's load and effectively configure them in advance. As a result, the total time for the integrated simulation tends to be longer than needed.

In order to solve the above problem, it is useful to dynamically configure the integrated simulators during simulation according to the current situation. From the viewpoint of fault tolerant, the dynamic configuration of distributed simulators is proposed (Welch and Purtilo 1997, Welch and Purtilo 1999). Those purposes are recovering simulators from unexpected troubles. Improvement of the simulation efficiency is not discussed very much. We propose a dynamic configuration method to reduce the total simulation time. In the proposed method, the load of each PC executing simulation and the traffic of data communication between PCs are periodically monitored, and the configuration of the distributed simulators are dynamically and automatically changed during simulation. It enables simulator users to easily simulate a large scale manufacturing system without considering the configuration of the simulators.

INTEGRATED SIMULATION

Simulators for PLC, CNC, robot, etc. have been developed in the manufacturing field. We can check the control program for the controller using the simulator. However, it is hard to verify the behavior of the whole manufacturing system which consists of lots of machines.

An integrated simulation environment, which can connect many simulators, is useful for testing a whole manufacturing system. Connecting various simulators is possible by communicating control signals and other information between them. Using only data communication does not accurately simulate an actual manufacturing system consisting of many machines, if the simulators are executed asynchronously. The simulation, which does not consider data communication delay between simulators, is not sufficient for system simulation. Moreover, integrated simulators are necessary to be distributed on several PCs, since an integrated simulation for manufacturing systems consists of many simulators. Therefore the

Proceedings 16th European Simulation Symposium
György Lipovszki, István Molnár © SCS Press, 2004
ISBN 1-56555-286-5(book) / ISBN 1-84233-106-x(CD)

integration of different kinds of simulators involves the following issues:

- Synchronization management
- Communication management
- Configuration management

Synchronization management

Figure 1 illustrates an example of simulation flow, in which three simulators are integrated asynchronously. The cycle times of the simulators are 100, 70, and 130, respectively. Each simulator executes a cycle of simulation, and exchanges data with connected simulators, then executes next cycle of simulation. In Figure 1, a vertical line is actual time, and an italic figure is logical time of the simulator. A rectangle is execution of simulation, and an arrow is a data flow between simulators.

In Figure 1, when the simulator2 (S2) sends data to the simulator1, 3 (S1, S3) at time 490, the S1 and S3 have already arrived at time 500 and 520, respectively. The simulation does not guarantee the timed consistency, since the S1 and S3 receive a past data.

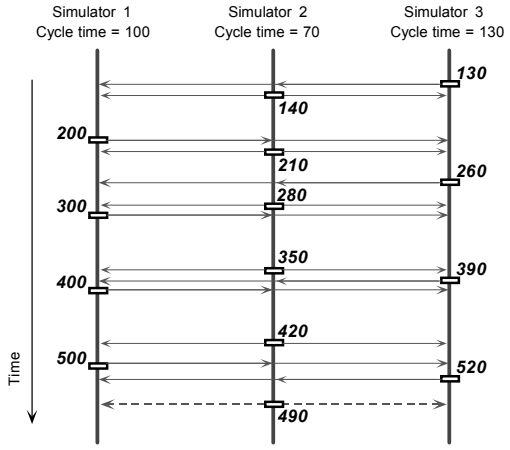

Figure 1: Simulation flow (asynchronous)

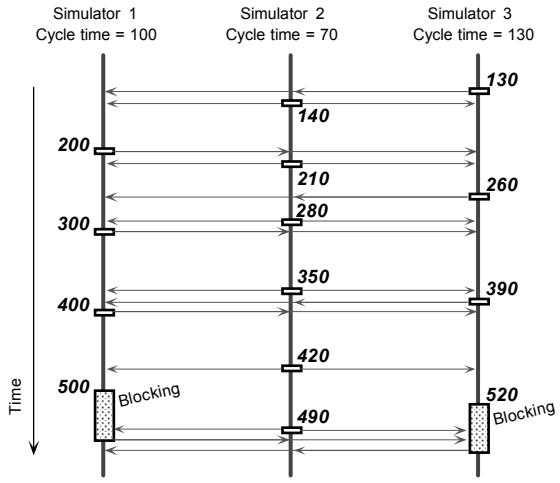

Figure 2: Simulation flow (synchronous)

Figure 2 illustrates a simulation flow in case of synchronous simulation. In this case, the S1 is blocked starting simulation at time 500, since the simulation of the S2 at time 490 is not finished. The S3 is also blocked at time 520 in the same way. It is possible to guarantee the timed consistency by synchronizing connected simulators.

Communication management

Integrated simulators have their I/O (input and output), and they are connected to others by exchanging data through the I/O. Synchronous data exchange guarantees the timed consistency.

There are many types of communication lines between equipment in manufacturing systems, for example a serial line, Ethernet and so on, and the data transfer speed depends on the line type (Figure 3). The simulation, which does not consider data communication delay according to the lines, is not sufficient for manufacturing systems.

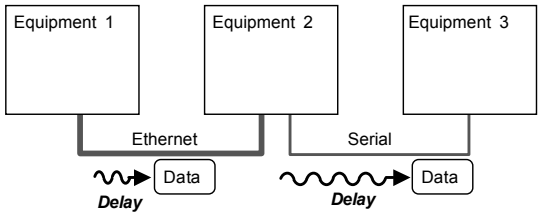

Figure 3: Data communication

Configuration management

Manufacturing systems consist of many pieces of equipment, and a large number of simulators are integrated and executed simultaneously when they are simulated. It is difficult to execute the all simulators on a PC, since the simulation needs a lot of processing power in general. Consequently, the simulators are necessary to be appropriately distributed on several PCs and be synchronously executed exchanging data for each others (Figure 4). Configuration management is important for a large scale of simulation like manufacturing systems.

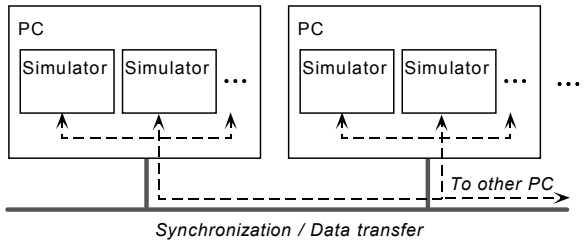

Figure 4: Configuration management of simulators

CONFIGURATION OF SIMULATORS

We already proposed the method of the synchronization management and the communication management in the

integrated simulation (Furusawa and Yoshikawa 2002). In this section, we discuss the configuration management.

Configuration in an integrated simulation

In the integrated simulation of manufacturing systems, the various types of simulators are integrated. Some of them require a lot of processing time for their execution. For example, a simulator of a motion controller has very short cycle time, and it usually needs high CPU power to simulate it. If many simulators, which require much processing, are executed on only a PC, its load becomes very high and the execution of the simulation needs a lot of time. As a consequence, the whole simulation does not advance effectively, even if all of the simulators on other PCs finish their simulation cycles.

To solve the above problem, the integrated simulators must be appropriately distributed to several PCs. It is necessary to estimate the capability and number of the PCs corresponding to the type and number of the simulators in advance, and to arrange the simulators to each PC.

Static configuration

In this section, the static configuration is explained, which defines the configuration of the used PCs and assignment of the simulators to the PCs before executing simulation.

Configuration of PCs
Optimizing the efficiency of the integrated simulation using the limited resources involves the following issues:

- Number of PCs
  The more simulators the integrated simulation executes, the more PCs it requires.
- Capability of PCs
  The more the simulators need the amount of computation, the higher performance PC the integrated simulation requires.
- Network between PCs
  The network configuration must be decided considering the location of the PC. For example, in case of executing a high load simulator on a high performance remote PC, the simulator might be connected to others via the internet.

Configuration of simulators
The assignment of the simulators to the PCs is necessary to be decided after the configuration of the PCs. The simulators should be appropriately distributed to the PCs in order to uniform their CPU load. In order to decide the desirable configuration, it is necessary to estimate the required CPU load, in advance, corresponding to the types of the simulators.

The communication traffic is another factor to decide the configuration. For example, when some simulators communicate their large data to each other, the communication traffic becomes very heavy if they are distributed to different PCs (Figure 5). The whole simulation time could be longer. In this case, executing the simulators on the same PC might improve the efficiency. The assignment of the simulators should be, therefore, decided considering the communication traffic between the simulators.

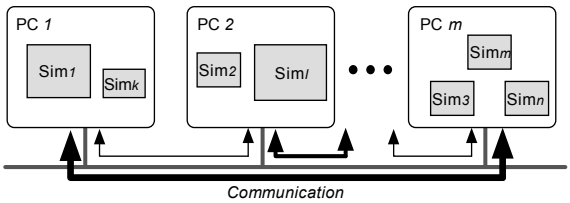

Figure 5: Communication traffic between simulators

Problems in static configuration

The static configuration is comparatively easy to implement a simulation environment, since it is usually defined before simulation and it is fixed during simulation. From the viewpoint of efficiency, however, there are problems in the static configuration as follows:

- Unbalance of the CPU load
  In the simulation for manufacturing systems, there are many types of simulators and the scale tends to be very large. Therefore, it is difficult to estimate the CPU load of the used PCs in advance. They are also changeable according to the condition. Under the static configuration, however, the distributed assignment of the simulators is fixed, and it could cause the unbalanced CPU load.
- Unbalance of the communication traffic
  It is difficult to estimate the communication traffic between simulators, and it is changeable as well. The static configuration could cause the unbalanced communication traffic.
- Inefficiency of the simulation execution
  The unbalance of the CPU load and communication traffic leads to decreasing the throughput of the PCs. As a result, the required time for simulation becomes longer, and the execution of the whole simulation becomes inefficient.
- Difficulty of deciding the configuration
  In case of the static configuration, the simulation user must estimate the specification and number of used PCs, and assign the simulators to them appropriately. It is so heavy a burden, and requires high skill and experience.

DYNAMIC CONFIGURATION

The static and fixed configuration causes the problems shown in the previous section. Dynamic configuration solves them. We propose a dynamic configuration method, which enables distributed simulators to dynamically change their configuration during the simulation.

The proposed dynamic configuration is realized by gathering the information on the CPU load of the used PCs and the communication traffic between them, and dynamically moving the simulators to the appropriate PC when the load and the traffic exceed a certain level during the integrated simulation.

Flow of dynamic configuration

In this section, we show the flow of the dynamic configuration in the integrated simulation. The flow is as follows:

Step1 Gather the information for reconfiguration
The information about the CPU load and the communication traffic between simulators is gathered. The information is used in order to judge whether the reconfiguration of the simulator is carried out.

Step2 Judge the reconfiguration
The information gathered in the Step1 is applied to the conditional expression of the reconfiguration, and then the necessity of the reconfiguration is judged.

Step3 Decide the reconfiguration
The new configuration is calculated. It improves the efficiency of the integrated simulation.

Step4 Stop the integrated simulation
The integrated simulation is stopped before its reconfiguration.

Step5 Execute the reconfiguration
In order to change the present configuration to the new one calculated in the Step3, the information on the simulator, for example the current status, is transferred to the reconfigured PC, and the simulator is prepared starting on the new configuration.

Step6 Restart the integrated simulation
The integrated simulation is restarted on the new configuration.

Conditional expression of reconfiguration

As we explained above, the conditional expression is used to decide the necessity of the reconfiguration. The necessity of the reconfiguration is judged based on the balance of the PC's load and the communication traffic. The conditional expression is lead by the following steps.

The required time for a cycle of simulation on a PC P_k, TS_k, is expressed by the following expression:

$$TS_k = \frac{\sum_{k=k1}^{km_k} L_k}{p_k} \tag{1}$$

where the simulators on the P_k are $S_{k1}, S_{k2}, \ldots, S_{km_k}$, and the amount of computation required for the simulation of S_k is L_k, and the amount of computation, which is able to be executed per unit time on the P_k, is p_k.

The required time for the data transfer on the P_k in a cycle of simulation, TC_k, is expressed by the following expression:

$$TC_k = \sum_{i=k1}^{km_k} \sum_{j=1}^{n} d_{ij} D_{ij} \tag{2}$$

where the amount of transferred data between S_i and S_j, is D_{ij}, and the required time for transferring a unit data between S_i and S_j is d_{ij}.

The required time for the integrated simulation on the P_k, T_k, is expressed by the sum of Equation (1) and (2).

$$T_k = TS_k + TC_k \tag{3}$$

The standard deviation of the required time on a PC, σ, is expressed by the following expression:

$$\sigma = \sqrt{\frac{1}{m} \sum_{k=1}^{m} (T_k - \overline{T})^2} \tag{4}$$

where $P_1, P_2, \ldots, P_m$ are PCs used for the integrated simulation, and $\overline{T}$ is the average of the required time $T_1, T_2, \ldots, T_m$.

Finally, the conditional expression of the reconfiguration is the following expression:

$$\frac{\sigma}{\overline{T}} \geq r \tag{5}$$

where r is a judging parameter ($0 \leq r \leq 1$). When the conditional expression (5) is true, the present configuration is judged not to be well-balanced and the reconfiguration is carried out. If the parameter r is small, the reconfiguration occurs frequently.

EVALUATION OF THE METHOD

In this section, we validate our dynamic configuration method by simulation.

Assumption

In order to validate our dynamic configuration method, we simulated it under various conditions. In the simulation, the number of the integrated simulators is 40 ($S_1, S_2, \ldots, S_{40}$), which is supposed to be components of manufacturing systems. The number of used PCs is 10 ($P_1, P_2, \ldots, P_{10}$). The method is evaluated changing the following three conditions:

- Performance of PCs
- Amount of computation for each simulator

- Amount of transferred data between simulators

The details of the conditions are shown in Table 1, Table 2 and Table 3. On the performance of PCs, two cases are evaluated as shown in Table 1. The all PCs have the same performance in the one case, and three types of performance in the other case. The required amount of computation for simulators and the amount of transferred data between simulators are randomly fluctuated inside the range shown in Table 2. The interval of their changes is also randomly fluctuated inside the range shown in Table 3.

In Table 1, the performance of PCs means the amount of computation, which can be computed per unit time. In Table 2, the amount of computation means the total amount of computation, which is required for a cycle of simulation of each simulator. All of the integrated simulators are supposed to communicate with each others, and the transferred data means the total amount of data sent and received per cycle of simulation. To simplify the problem, the two conditions for the amount of computation and the transferred data are combined, though they are independent conditions. And the required time for data transfer between two simulators on the different PCs is supposed to be 0.001 sec/KB, and the one between two simulators on the same PC is supposed to be 0.00001 sec/KB. And the cycle time of the every simulator is supposed to be 0.1 sec. The judging parameter r is 0.1, and the reconfiguration is judged at 60 sec interval. The simulators are initially assigned based on the performance of the PCs.

Table 1: Condition (PC performance)

Condition	Performance of PCs
Cp1	400 [10 PCs]
Cp2	1000 [1 PC], 400 [4 PCs], 200 [5 PCs]

Table 2: Condition (simulation load)

Condition	Amount of computation	transferred data (KB)
Cc1	1	0.01
Cc2	Min 1, Max 2	Min 0.01, Max 0.02
Cc3	Min 1, Max 5	Min 0.01, Max 0.05
Cc4	Min 1, Max 10	Min 0.01, Max 0.10
Cc5	Min 1, Max 20	Min 0.01, Max 0.20
Cc6	Min 1, Max 40	Min 0.01, Max 0.40
Cc7	Min 1, Max 80	Min 0.01, Max 0.80

Table 3: Condition (change interval)

Condition	Change interval (sec)
Ci1 (short)	Min 60, Max 120
Ci2 (long)	Min 480, Max 960

From the view of simulation efficiency, the configuration minimizing $\max(T)$ in consideration of all PCs and simulators is the optimal one. However, the combination of $P_1, P_2, \ldots, P_m$ and $S_1, S_2, \ldots, S_n$ is m^n, and it costs too much time to evaluate all patterns and to optimize them. In our evaluation, therefore, we selected two PCs, whose $\max(T)$ is the largest and the smallest, and then we optimized the assignment of the simulators on these two PCs.

Evaluation results

We evaluated the required total simulation time under various conditions. In our evaluation, the required total time using our dynamic configuration is compared with one not using it. The required total time includes the time for computation of simulation and transferring data, and the overhead for reconfiguration of simulators, for example moving simulators, restarting them, and so on, is not included. It, however, has not so large influence on the efficiency of the simulation, since the reconfiguration interval is sufficiently long. We show the results of simulation. The duration of the simulation is 10000 sec (almost 3 hours).

Firstly, in the cases using the same performance PCs, the results of the simulation are shown in Figure 6 and Table 4. The case of short fluctuation interval and the case of long one are evaluated. The improvement of the required total simulation time is not confirmed, when the fluctuation range of the amount of computation and the transferred data is small. The larger the range is, however, the larger the reduction of the total simulation time is.

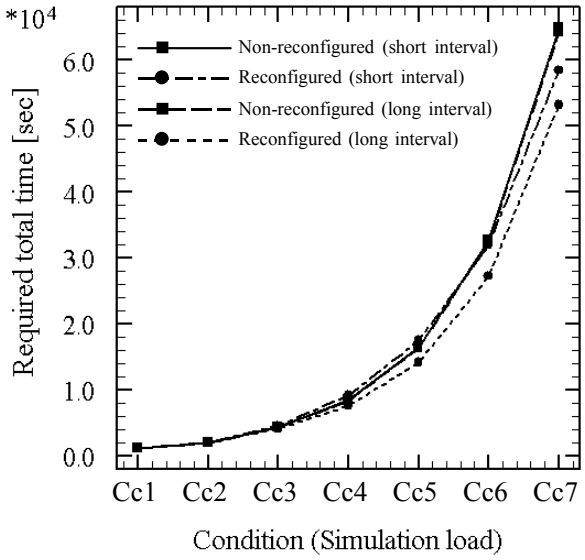

Figure 6: Required total time (condition Cp1)

Secondly, in the cases using the different performance PCs, the results of the simulation are shown in Figure 7 and Table 5. Unlike the first cases, the improvement of

the required total time is confirmed, even when the fluctuation range of the amount of computation for simulating and the amount of transferred data is small. The larger the range is, the larger the reduction of the required total time is, in the same way as the first cases. And the reduction rate is twice as high as the first cases.

Table 4: Reduction rate of total time[%] (condition Cp1)

Interval	Cc1	Cc2	Cc3	Cc4	Cc5	Cc6	Cc7
Ci1 (short)	0.0	0.0	-4.2	-8.9	-6.3	1.8	9.9
Ci2 (long)	0.0	0.0	3.8	9.0	13.3	15.7	17.2

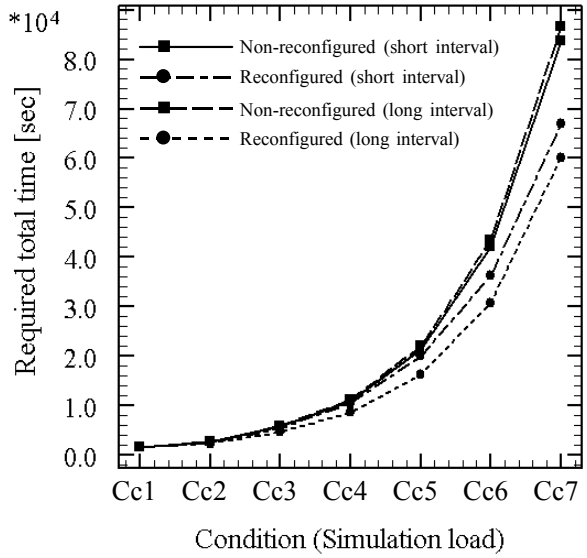

Figure 7: Required total time (condition Cp2)

Table 5: Reduction rate of total time[%] (condition Cp2)

Interval	Cc1	Cc2	Cc3	Cc4	Cc5	Cc6	Cc7
Ci1 (short)	0.0	6.5	4.4	4.0	6.4	13.8	20.0
Ci2 (long)	0.0	13.0	20.6	22.7	26.1	29.2	30.6

CONCLUSION

We proposed the dynamic configuration method of the integrated simulation for manufacturing systems. The proposed method can automatically adjusts the configuration of the large scale integrated simulation, and the user of the simulator can efficiently carry out the complicated simulation without considering the configuration, using the limited resources. We evaluated our method and it is confirmed that the required simulation time is drastically reduced, especially when the range of fluctuation of simulation load is large, though the overhead of the reconfiguration is not included in the evaluation.

In this paper, our method is evaluated by only simulation, and the overhead of the reconfiguration is not discussed. It should be also evaluated in the real situation, in order to examine the effect of the overhead and verify the practicality of our method.

REFERENCES

Carothers, C.D.; Fujimoto, R.M.; Weatherly, R.M.; and Wilson A.L. 1997. "Design and Implementation of HLA Time Management in the RTI Version F.0." *In Proceedings of the 1997 Winter Simulation Conference*, 373-380.

Dahmann, J.S.; Fujimoto, R.M.; and Weatherly, R.M. 1997. "The Department of Defense High Level Architecture." *In Proceedings of the 1997 Winter Simulation Conference*, 142-149.

Defense Modeling and Simulation Office. 1998. *High Level Architecture Interface Specification, v1.3*.

Furusawa, K. and Yoshikawa, T. 2002. "Synchronization Mechanism in Integrated Simulation for Manufacturing Systems." *In Proceedings of the MED2002*.

Welch, D.J. and Purtilo J.M. 1997. "Using Compensating Reconfiguration to Maintain Military Distributed Simulations. " *Proceedings of the 1997 Winter Simulation Conference*, 961-967.

Welch, D.J. and Purtilo J.M. 1999. "Building Self-Reconfiguring Distributed Simulations Using Compensating Reconfiguration." *The Journal of Defense Software Engineering*, 20-23.

AUTHOR BIOGRAPHIES

KOICHI FURUSAWA was born in Osaka, Japan and went to the Osaka University of Japan, where he studied computer science and obtained his B.E. degree and M.E. degree in 1993 and 1995. Then, he works for the Mitsubishi Electric corp. where he is now a research engineer in the controller group of the advanced technology R&D center in the field of a programmable logic controller. His e-mail address is Furusawa.Koichi@wrc.melco.co.jp.

KAZUSHI OHASHI was born in Osaka, Japan and went to the Osaka Prefecture University of Japan, where he studied robotics and obtained his B.E. degree in 1988 and went to the Osaka University of Japan, where he obtained M.E. degree in 1990. Then, he works for the Mitsubishi Electric corp. where he is now leading a controller engineering unit in the controller group of the advanced technology R&D center. His e-mail address is Ohashi.Kazushi@wrc.melco.co.jp.