Simulation Objects in Distributed Environment

Hristo Valchanov [1]

Abstract – **Simulation is a modern approach used for modelling large complex systems and understanding their behaviour. The parallel discrete event simulation (PDES accelerates the modeling process by distributing it among a number of processors. Local area networks are an available platform for PDES. The distributing of the simulation entities over the processors is very important for the performance of the simulation process. This paper presents an approach for mapping the simulation objects over distributed environment - network of workstations.**

Keywords – **PADS, PDES, Mapping, Simulation objects.**

I. INTRODUCTION

PDES accelerates the modeling process by distributing it among a number of processors. With PDES, the modeled system is presented as a set of sub-systems simulated by a number of simulation objects (SO). The SO communicate with one another by exchanging time-stamped messages for occurring events. The simulation correctness requires that the events be processed in the order of their occurrence in time. Special synchronizing protocols are used to ensure the right order of processing [1].

The simulation objects may be implemented as separate independent processes. Such implementation, however, is ineffective from the point of view of the high system overhead on switching of the processes context by the operating system (OS). On the other hand, the communications between the processes in a same computer node is implemented by the OS IPC messages mechanism and has approximate complexity as the inter-computer network communications. By these reason the speed of simulation is largely reduced. Simulation effectiveness improvement can be achieved by aggregating the simulation objects into a cluster. Each cluster will perform as an independent process within a computer node. Its purpose is to carry out scheduling of its simulation objects and ensure communication with the other clusters within the network.

This paper presents an approach for mapping of the SO into clusters over distributed simulation environment based on a network of workstations.

II. RELATED WORK

Distribution of SO over computing nodes is very important for the efficiency of the overall process of simulation. Numerous systems for PDES [3] provide such control of distribution, which require the user to map manually SO to the appropriate physical processors. This approach appears to be inefficient for simulation of models, containing many SO with high intensity of interaction. For such models it is very difficult for the user to determine the interaction between the components of the model, as well as to estimate the optimal configuration of distribution of SOs to the physical processors. There is a need to automate the process of mapping the appropriate SO to the computing nodes.

The selection of a method for distribution of the components of the simulated system into groups (clusters) has an important role for the efficiency of the overall process of simulation. One method is to distribute SO equally into clusters. This method is easy to implement and it is based on the equalization of the computing overhead in separate nodes. However, such mapping does not take into consideration the interaction between SO in the simulated system. This fact has very great influence over the simulation performance in case of simulation of complex dynamical system over distributed environment.

Another method is based on a representation of the simulated system as a graph, which is to be distributed by means of algorithms for graphs partitioning. By this method, the vertices of the graph represent the individual components of the real system, while the edges of the graph represent the interaction between the components. The edges have assigned weights that representing the amount of communication between the components. Applying algorithms for graphs partitioning results in division of the graph into relatively equal parts, thus minimizing the total communication between these parts. The method is also relatively easy to implement, because the problem of graph partitioning is well known in the graphs theory [4].

An important problem for this method is the manner of building of the graph of interaction between SO. A possible solution is to include a specific analysis within the compiler of the simulation language. This method allows generation of the information for the interaction between instances of simulation classes during the stage of compiling. This analysis, however, would be extremely complicated to implement because it is necessary to consider the dynamics of exchange between the components of the simulated system. In case of a dynamic creation of instances of the SO the complexity will increase.

Another solution is based on the *critical path analysis* in the process of simulation [3]. The key concept of this method is that if the graph of the parallel program execution (sequence of events) is known, then the critical path gives the least possible time for execution of simulation. The analyzers, presented in the literature [3], require completion of the whole process of sequential simulation for carrying out the analysis of the critical path (i.e. *post-mortem* analysis). This is applicable to sequential simulation, executed within

[1]Hristo Valchanov is with the Computer Science & Engineering Dept. at Technical university of Varna. 1, Studentska Str., 9010, Varna, Bulgaria, E-mail: hristo@tu-varna.bg.

reasonable period of time. In case of simulation models, involving large number of SO and events such requirement will result in too long execution. At the same time, such analyzers require amounts of memory and disc space proportional to the number of the simulated events and communication operations.

III. THE APPROACH

Our approach to the building of a graph of interaction of SO is based on preliminary sequential simulation, combined with dynamic analysis of the interactions between the components of the simulation model. The approach consists of two phases. During the first phase the graph of interactions between SO is created. The second phase includes partitioning the formed graph into clusters.

A. Formation the graph of interactions

For the formation of the graph of interactions we use an experimental sequential simulator with integrated analyzing component [5]. Its purpose is to analyze the interaction between SO in the preliminary execution of a simulation program. Significant difference between analyses described in the literature and presented method is that it focuses solely on the interaction between SO and not on the sequence of simulation events.

As a result of the first phase the AC builds an interaction graph $G(V,E)$ between SO, where $V=\{v_i\}, i=1..N$ is the set of graph vertices. This set corresponds to the set of the simulation objects $O=\{o_i\}, i=1..N$. The set of events $E=\{e_i\}, i=1..M$ is represented as edges of the graph and each edge introduces the interaction between the SO. Two vertices v_k and v_q are connected with edge e_{kq} if the corresponding SO O_k and O_q exchange messages about happened events during the simulation.

Formation of the graph of interactions $G(V,E)$ may be presented as a two-steps process: during the first step, the communication pairs between SO are formed, and during the second step information about the exchange of messages between them is collected. The grounds in this regard are the following. Upon generating simulation models, the user initially forms on the basis of specific language structures, the logical topology of the connections between SO. After starting the simulation, communication exchange between SO begins, reflecting the scheduling of the simulation events. While the number of events may vary during the whole process of simulation, formation of communication pairs is made yet at the beginning. The primary task is to determine the time T_{CP}, required for the formation of all communication pairs (P_{CP}) during the process of sequential simulation. After formation of P_{CP}, the process of sequential simulation (T_{sim}) continues for a specified period of time - $T_{end}=\lambda * T_{CP}$ during which data about the amount of communications are collected. The value of the parameter λ indirectly determines the accumulation of information exchange between communication pairs.

The detailed explanation of this phase is presented in [5].

B. Partitioning the graph of interactions

Formally the problem of graph partitioning is defined as follows: a graph $G(V,E)$ with set of vertices V and set of edges E is given. Let V be divided into k subsets $V_1,V_2,...,V_k$ such that:

1. $V_i \cap V_j = \Phi, \forall i \neq j$, where Φ is an empty set;
2. $\bigcup_{i=1}^{k} V_i = V$;
3. $|V_i| = \frac{|V|}{k}$;
4. The number of edges connecting vertices from different subsets is minimal.

The conditions 1 and 2 determine the division of the graph with number of vertices $|V|$ into k non-overlapping sub-graphs. The condition 3 determines that the number of vertices in individual sub-graphs must be equal.

For partitioning of the formed graph G the combinatorial multilevel-based method named *Multilevel k-way* is applied [4]. The choice of the algorithm *Multilevel k-way* is based on the following considerations. First, it incorporates the optimization criterion, very appropriate for simulation in distributed computing environment. The criterion is minimization of the general communication exchange between the computing nodes. Compared to other methods of study, it allows precise graph partitioning at comparatively low computing expenses. Secondly, the algorithm is implemented on the basis of the library METIS [4], which is available as an open source and enables the use of API functions in the applications. The execution of the algorithm *Multilevel k-way* on the formed graph generates a map of SO distribution by computing nodes.

Fig. 1 displays the process of distribution of SO between computing nodes.

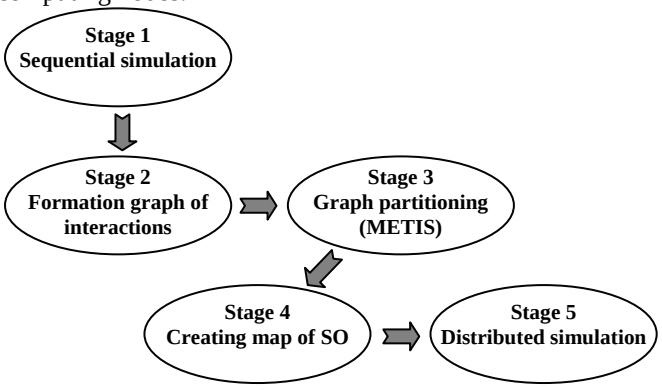

Fig. 1. Process of SO mapping

After completion of the sequential simulation (stage 1), data are generated for the formed graph of interactions $G(V,E)$

and these data serve as input data for the METIS program (stages 2 and 3). The result is a file, containing the map of distribution of SO between the relevant computing nodes (stage 4). Once the map is generated, the distributed simulation may start (stage 5). When a new SO must be created during the simulation, the run-time system uses information from this map. As a result, the SO is created on a particular node.

C. Experimental evaluation and results

Experimental studies has been carried out on the basis of the benchmark test PHOLD [1], [2] using SIMOPAL distributed simulation environment [6], Fig2. This test is widely used for assessment of the distributed simulation performance.

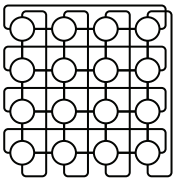

Fig. 2. PHOLD example (N=16)

The test model contains N objects, connected in 2D toroidal network and E events, exchanged between the objects. The dependence has been studied between the number of communication pairs P_{CP} and the number of the processed events during sequential simulation.

Experiments are carried out in three groups. For the first group, the process of sequential simulation is waited until completion. For the second group of experiments simulation is terminated upon achievement of 90% of the total duration T_{sim}. For the third group the proposed method is applied, whereas simulation is terminated upon reaching the time $T_{end} = \lambda * T_{CP}$ (these times correspond to the processing of a specified number of events). Experiments are carried out at different values of λ.

Comparative assessment is made by the following methods - for each computing node is formed a set Ω of SO, which are assigned to it as a result of the complete process of simulation, and the set $\tilde{\Omega}$, containing SO, assigned to it as a result of the proposed method,

$$\bigcup_{j=1}^{n} \Omega_j = \bigcup_{i=1}^{n} \tilde{\Omega}_i = N, \text{ where} \qquad (1)$$

N is the number of all SO in the model, and **n**- the number of computing nodes.

For each set $\tilde{\Omega}_i, i = 1..n$ is determined the maximum ratio $\Psi_i^{\max}, i = 1..n$ (in percentages) of the coincidences of its components with each set $\Omega_j, j = 1..n$. For each group of distribution by nodes is determined the average maximum ratio

$$\overline{\Psi}_i = \frac{\sum_{j=1}^{n} \Psi_j^{\max}}{i} [\%], i = 1..n \qquad (2)$$

The aim is that on the basis of experimental studies λ is determined at which the value $\overline{\Psi}_i, i = 1..n$ will be highest.

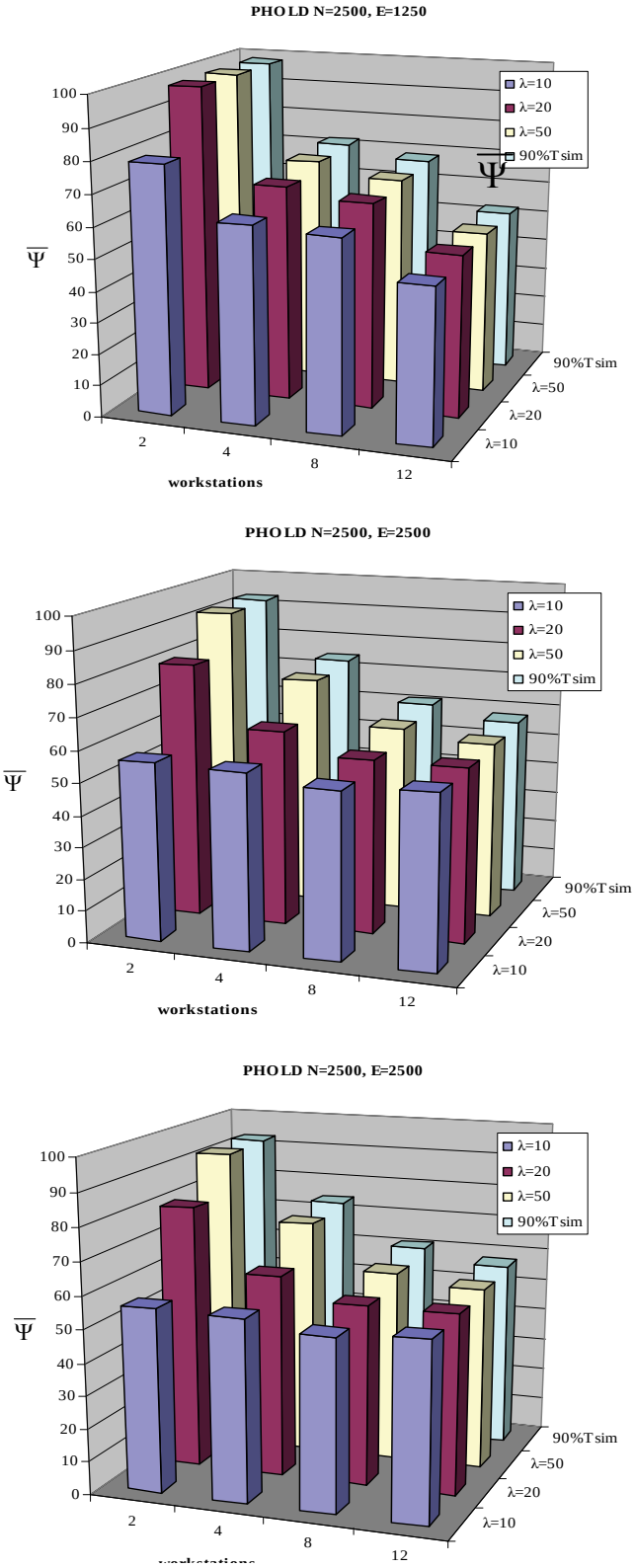

Fig. 3. Obtained results

Fig.3 shows the results from carried out experimental studies. Experiments are carried out by increasing the number of simulated events E. As a number of objects we use N=2500 because with that value the highest performance for network with 12 workstations is achieved. Comparative assessment is made with regard to the results, obtained upon achievement of 90% of the total duration of simulation T_{sim}.

As seen from the charts, with increasing number of simulated events we observed some reduction in the percentage of coincidences. This is due to the fact that a larger number of events needed more time to accumulate information about the communication exchange between communication pairs.

Increase the number of computing nodes **n** also reduces $\overline{\Psi}$. This is a normal consequence of increasing the number of communication channels between communication pairs. It should be noted that this reduction is lower when simulation uses large number of computing nodes. That indicates the correctness of *Multilevel k-way* choice in terms of its resistance to growing the size of the simulation.

As λ increases, it is logical that coincidence of sets $\overline{\Psi}$ increases too, due to the increase of information about communication exchange between simulation objects. Very important result is the level of coincidences upon achievement of 90% of the total time for sequential simulation T_{sim}. As it is evident from the graphics, regardless of the number of computing nodes and the number of simulated events, this proportion is within very close limits to the results in the cases of $\lambda > 20$. On this basis we may assume that choosing λ with values higher than 20, will allow obtaining distribution of SO over the computing nodes which are good enough upon the initial start of distributed simulation

IV. Conclusion

The mapping of simulation objects to processors is extremely important characteristic of systems for PDES to balanced load and inter-processor communication. Most of the above mentioned approaches leave this task to programmers or require completion of the whole process of sequential simulation for carrying out the analysis.

In this paper we proposed a new approach to partitioning of a graph of interaction of SO. Its key concept is based on preliminary sequential simulation, combined with dynamic analysis of the interaction between the components of the simulation model. Significant difference between analyses described in the literature and presented method is that it focuses solely on the interaction between SO and not on the sequence of simulation events The experimental results show that the proposed approach has efficiency for simulation models, characterized by high dynamics of scheduled events between SO.

The purpose of a future work will be to study the period, necessary to prolong the process of sequential simulation after reaching the required limit of communication pairs P_{CP}. Another direction will be a study of possibilities for integration of the proposed approach with dynamic load-balancing of SO between processors during the simulation.

Acknowledgement

The work presented in this paper was supported within the project BG 051PO001-3.3.04/13 of HR Development OP of the European Social Fund 2007-2013.

References

[1] J. Banks, J. S. Carson, B. L. Nelson, *Discrete-Event System Simulation* (5th ed.), Upper Saddle River, Prentice Hall (2009).

[2] R. Fujimoto, "Parallel Discrete Event Simulation", Communications of the ACM, vol. 33, pp. 41-52, 1990.

[3] Z. Juhasz, S. Turner, M. Gerzson. "A Trace-based Performance Prediction Tools for Parallel Discrete Event Simulation", Proc. Applied Informatics, vol. 3, pp. 338-343, 2002.

[4] K. Schloegel, G. Karypis, V. Kumar, "Parallel *Multilevel* Algorithms for Multi-constraint Graph Partitioning", LNCS, vol.1900, pp. 296—310, 2000.

[5] H. Valchanov, N. Ruskova, D. Genov, N. Nikolov, "Partitioning Parallel Discrete Event Simulation", Proc. of CompSysTech08, pp. IIIA.5-1 IIIA.5-6, 2008.

[6] H.Valchanov, N.Ruskova, T.Ruskov. "Distributed Simulation over Network of Workstations", Proc. of CompSysTech 2006, V.Tarnovo, IIIB.25-1, IIIB25-6, 2006.