Simulation Modelling Practice and Theory 52 (2015) 123–134

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

Real-time load balancing scheduling algorithm for periodic simulation models

Yulin Wu [1], Xiao Song [1,*], Guanghong Gong

School of Automation Science and Electrical Engineering, Beihang University, No.37 Xue Yuan Road, Hai Dian District, Beijing 100191, China

ARTICLE INFO

Article history:
Received 1 July 2014
Received in revised form 23 October 2014
Accepted 7 January 2015
Available online 7 February 2015

Keywords:
Real-time system
Scheduling
Time-stepped simulation
Jitter

ABSTRACT

A scheduling algorithm is crucial for real-time simulations because it guarantees that each model meets its deadline. Traditional online real-time scheduling algorithms such as Earliest Deadline First (EDF) introduce a high overhead when scheduling a large number of models. In this paper, a new algorithm called time-stepped load balancing (TLS) is proposed to address the real-time execution of a model set in a time-stepped simulation. A load balancing schedule table is generated before a simulation and rebalanced at runtime to dynamically schedule the changed model set. This table is organized by the execution periods of the models and balanced according to the load of each time step. Moreover, the slack time is distributed evenly among the steps to improve the real-time reliability. An extension to the algorithm for a multi-core environment is further studied to address those models with long execution times. Experimental results show that our scheduling algorithm outperforms the classical EDF approach. The highest performance improvement of TLS over EDF reaches 3–4% in terms of saving processor resources, and the jitter is about 4 times less when 90 entities are employed in a typical tank combat simulation scenario.

1. Introduction

Human-in-the-loop simulations connect computer models with human actors. Thus, it would be unrealistic if the simulation speed was slower than real-time. The fidelity of a real-time simulation depends not only on the performance but also on the timing simulation results that are produced [1]. Sometimes a constructive simulation without a real human must also run in real-time to avoid performance degradation [2].

Real-time means the ability to process tasks in time to meet deadlines. Real-time simulations can be divided into two types according to the seriousness of the time requirements: soft real-time and hard real-time [3]. In the former, missing the deadline occasionally is acceptable, e.g., video telephone, and it is concerned more with the statistics rather than rigorous adherence. In the latter, missing a deadline only once may lead to an error or a disaster, e.g., flight control system. Most scheduling algorithms such as EDF (Earliest Deadline First scheduling) can be applied or revised to meet the requirements of both soft and hard real-time simulations. Meanwhile, the hardware, OS (operating system), and simulation middleware also play parts in real-time [4]. Although hardware performance has been continuously improved, a highly efficient real-time

[*] Corresponding author.
E-mail addresses: wyl@asee.buaa.edu.cn (Y. Wu), songxiao@buaa.edu.cn (X. Song).
[1] The first two authors contributed equally to this work.

http://dx.doi.org/10.1016/j.simpat.2015.01.001
1569-190X/© 2015 Elsevier B.V. All rights reserved.

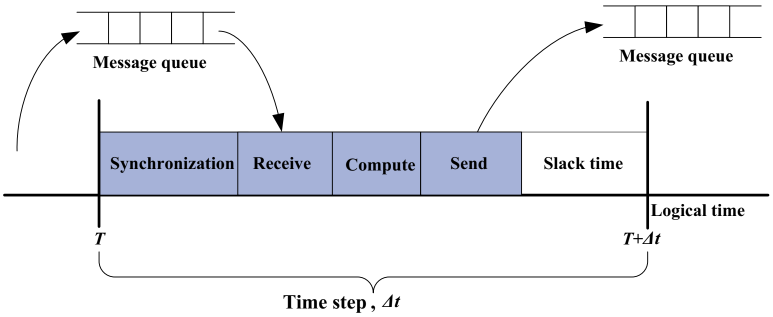

Fig. 1. Time step in simulation.

scheduling algorithm is still a challenging problem for simulating complex systems, because the number of simulation models is also increasing rapidly along with the user requirements.

A scheduling algorithm is essential for real-time simulations because it determines the appropriate order of model executions and guarantees that each model is able to meet its execution deadline. Scheduling algorithms can be divided into static and dynamic methods [3]. A static method determines all the execution orders before running tasks. A dynamic method only determines the current task to be executed at runtime, e.g., EDF assigns the highest priority to the task with the earliest deadline, which is calculated dynamically. Normally, a static method is also called an offline method, and a dynamic method is called an online method.

Time-stepped simulation has been common in practice. For example, many military simulations based on a High Level Architecture (HLA) [5] request a time advance with a time step. In a large-scale simulation, the time step is controlled by the simulation framework that manages the models. Some models in these simulations must be executed periodically, and they usually consume the majority of the computing resources. The period refers to how often the model runs and is set carefully to provide the required precision. To meet the deadline, the scheduler often needs to know the worst execution time of the model [3].

This paper investigates the scheduling problem on the model level with a time-stepped simulation. We address the problem by combining the static scheduling and dynamic adaptive methods. The initial models are assigned to a scheduling table, which is then modified in line with newly created or destroyed models. The scheduling table plans the running instances of each time step according to the model periods. Therefore, the models to be run can be determined directly at runtime. The main contribution is the proposed time-stepped load balancing scheduling algorithm that maintains the scheduling table.

This paper is organized as follows. In Section 2, the works related to real-time scheduling algorithms, as well as the real-time simulation middleware, are reviewed. A time-stepped simulation is illustrated in Section 3. Section 4 presents the real-time scheduling algorithm we proposed for the simulation models. An analysis of the results of an experiment with the algorithm is given in Section 5. The conclusions are presented in Section 6.

2. Related works

Processor utilization is an important performance indicator for algorithms [6]. EDF can reach 100% utilization (in non-preemptive mode) and accommodate dynamic tasks. EDF has been the most commonly used algorithm [7] and has been optimized for different environments. Cheng et al. [8] presented a semi-preemptive EDF to handle the network service flow. The task priority was optimized to reduce the preemption and enhance the service delay performance. EDF can also be easily applied to a simulation system. Recent research on the real-time scheduling of HLA tasks with the double-earliest-deadline-first (D-EDF) algorithm was presented [9] to deal with both periodic and aperiodic models; however, the performance was not verified. Stavrinides and Karatza [7] used a bin packing technique to enhance multiple real-time scheduling algorithms,

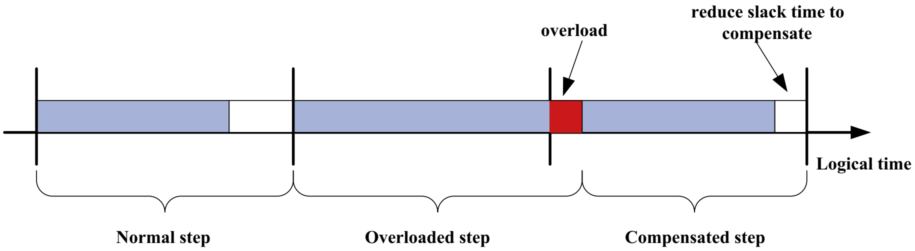

Fig. 2. Time compensation between two steps.