Co-simulation of dynamic systems in parallel and serial model configurations†

Trevor Sweafood1 and Hwan-Sik Yoon‡,*

1General Motors, Milford, MI, USA
2Department of Mechanical Engineering, The University of Alabama, Tuscaloosa, AL 35487-0276, USA

(Manuscript Received November 27, 2012; Revised July 5, 2013; Accepted July 25, 2013)

Abstract

Recent advancements in simulation software and computation hardware make it realizable to simulate complex dynamic systems comprised of multiple submodels developed in different modeling languages. The so-called co-simulation enables one to study various aspects of a complex system with heterogeneous submodels in a cost-effective manner. Among several different model configurations for co-simulation, synchronized parallel configuration is regarded to expedite the simulation process by simulating multiple submodels concurrently on a multicore processor. In this paper, computational accuracies as well as computation time are studied for three different co-simulation frameworks: integrated, serial, and parallel. For this purpose, analytical evaluations of the three different methods are made using the explicit Euler method and then they are applied to two-DOF mass-spring systems. The results show that while the parallel simulation configuration produces the same accurate results as the integrated configuration, results of the serial configuration show a slight deviation. It is also shown that the computation time can be reduced by running simulation in the parallel configuration. Therefore, it can be concluded that the synchronized parallel simulation methodology is the best for both simulation accuracy and time-efficiency.

Keywords: Co-simulation; Dynamic system simulation; Parallel simulation; Serial simulation

1. Introduction

Computer simulation of physical systems has been widely adopted in many science and engineering disciplines. Due to the increasing affordability of high computation power and significant improvements in high-precision simulation tools, it is now possible to simulate complex dynamic systems with very high accuracy. When compared to the conventional product design and development process, the design of complex dynamic systems through computer simulation can be done with a fraction of the prototype development cost. Thus, the so-called virtual prototyping is now becoming an important component in the development of complex engineering systems such as aircraft, ground vehicles, and electrical systems.

As simulated systems become larger and more complicated, sometimes it is necessary to develop subsystem models independently at different times. This type of modular system modeling allows partial upgrade or modification in a large complex model so that other parts of the system model can be reused to save development time and cost. The subsystem models or submodels are sometimes written in different languages due to the different strengths and capabilities of various software packages. In order to overcome the difficulty with simulating dynamic systems with multiple submodels written in different languages, co-simulation is currently being studied by an increasing number of researchers. Co-simulation is a simulation methodology whereby multiple heterogeneous submodels are simulated together by exchanging information or simulation results in a certain predefined manner. In this way, complete system-level simulation of a large complex system can be possible with unprecedented detail and accuracy.

Recent efforts toward co-simulation strategies have prevailed in automotive applications. Kim et al. conducted a co-simulation of ADAMS and MATLAB models to evaluate vehicle stability control algorithms for a four wheel drive hybrid electric vehicle [1]. Mancosu and Arosio co-simulated a multi-body vehicle model in ADAMS and vehicle subsystem models in MATLAB/Simulink [2]. Engine response was also explored by co-simulating engine component submodels created by the most suitable software program [3]. Chassis and suspension systems have also provided opportunities for realizing co-simulation techniques [4-6]. Datar et al. co-simulated a full vehicle model using ADAMS and PSAT by means of a connection between models created in the MATLAB/Simulink environment [7]. The overall dynamic vehicle behavior
was to be estimated by analyzing the interaction of the powertrain and the vehicle. The co-simulation of a GT-Power engine model and a MATLAB/Simulink optimization algorithm was also undertaken [8]. Prevalent circuit technology efforts survey hardware/software co-simulation, which ensures that the hardware and software components in a system operate in a complementary manner [9]. Yi et al. applied a time synchronization technique to an MPSoC and compared the results to those obtained from a SystemC simulation framework [10]. Their co-simulation framework proved to be faster and operated with less than 5% error. Operating systems and embedded systems have also been studied for hardware/software co-simulation improvements [11-13].

Co-simulation of multiple submodels can be conducted in various configurations. Parallel simulation exploits hardware-level multitasking capabilities of multicore CPUs or networked computation environments. Whereas the simulation of multiple submodels is done one at a time in a serial configuration, in a parallel configuration, all of the submodels are run concurrently on different CPUs or CPU cores to reduce the overall computation time significantly. Although it is well known that parallel computation reduces the overall computation time, it is not clear whether the simulation results from the parallel configuration are the same as those obtained from serial computation. In light of this uncertainty, an important question is naturally raised. If these two different configurations produce different results, which one produces more accurate results? This paper seeks to answer this question by studying and comparing the performance of three different co-simulation configurations - single integrated, serial, and parallel - for two simple, mechanical mass-spring systems. The integrated configuration in this study is used as a reference system for the performance evaluation of the serial and parallel configurations. In addition to the simulation accuracy, a comparison of the computation time for the three different co-simulation configurations is made in this paper. Overall, the contribution of this paper is to illustrate the advantages of running co-simulation in the parallel configuration, using simple examples. It is designed to be helpful for practicing mechanical engineers.

This paper is organized as follows. In section 2, principles of the serial and parallel simulation methods are described, and three different simulation model configurations are defined using block diagrams in section 3. In section 4, the three configurations are applied to a two degree-of-freedom mechanical system and the simulation accuracies are analytically studied and compared using the explicit Euler method. The numerical simulation results are presented and interpreted in section 5, along with another set of simulation results from a different two degree-of-freedom system. A conclusion of this study will follow in section 6.

2. Serial vs. parallel computation

A physical multi-component system evolves incrementally in time, with each component interacting with others. This time domain evolution occurs simultaneously in all parts of the system by exchanging mass, force or energy between different components. In a co-simulation framework, this type of system can be simulated in a serial manner where each subsystem is simulated one after another as shown in Fig. 1. Note that there exists a time slot for data transfer between adjacent simulation processes, or subsystem models. However, this internal data transfer occurs asynchronously after the completion of each submodel. Two distinct simulation parameters related to time can be defined. The simulation time is the time period over which a system is simulated. On the contrary, the computation time is the actual amount of real time that has elapsed as the simulation progresses on a specific computer. In the serial configuration, the overall computation time depends on the number of submodels and the average time it takes to advance each submodel by one step size.

The concept of parallel co-simulation promotes the idea that multiple submodels can be simulated concurrently in a synchronized parallel manner. The nature of the dynamic models used for the simulation is such that each submodel requires data produced by other submodels at the previous time step in order to advance to the next time step, implying that data exchange must occur synchronously. Thus, at each time step, the response of a certain submodel can be calculated independently of the other submodels, allowing it to be done in a parallel manner. This process is continued until the simulation runs to completion. The actual time required to complete one simulation time step is limited by the submodel that runs slowest. The overall computation time is calculated by adding all the computation time for the longest-running model within each time step and the time for data exchange. This algorithm can be better explained by Fig. 2.

For this illustration, it is assumed that a given system model is partitioned into three submodels that are running on three