Journal of Computational Science 5 (2014) 882–890

Contents lists available at ScienceDirect

Journal of Computational Science

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

On the definition of a computational fluid dynamic solver using cellular discrete-event simulation

Michael Van Schyndel[a], Gabriel A. Wainer[a,*], Rhys Goldstein[b], Jeremy P.M. Mogk[b], Azam Khan[b]

[a] Department of Systems & Computer Engineering, Carleton University, Center for Visualization and Simulation (VSIM), Ottawa, ON, Canada
[b] Autodesk Research Toronto, ON, Canada

ARTICLE INFO

Article history:
Received 30 October 2013
Received in revised form 1 May 2014
Accepted 1 June 2014
Available online 9 June 2014

Keywords:
Computational fluid dynamics
Cellular Automata
Discrete event system
Biomechanical simulations

ABSTRACT

The Discrete Event System Specification (DEVS) has rarely been applied to the physics of motion. To explore the formalism's potential contribution to these applications, we need to investigate the definition of moving gases, liquids, rigid bodies, and deformable solids. Here, we show how to use Cell-DEVS to analyze the movement and interactions of fluids using computational fluid dynamics (CFD). We describe a set of rules that produce the same patterns as traditional CFD implementations. We present the inner workings of the CFD algorithm, the incorporation of solid barriers, and the adoption of variable time steps within the context of biomechanical simulations.

© 2014 Elsevier B.V. All rights reserved.

1. Introduction and motivation

The Discrete Event System Specification (DEVS) formalism, described in [1], has two properties that facilitate a scalable approach to simulation development. First, DEVS has been shown to be exceptionally general when compared with other modeling formalisms [2], allowing it to be applied to a wide range of simulation methods. Second, multiple DEVS models are easily coupled to represent more complex systems, even if the component models exhibit different time advancement patterns. A key principle of DEVS is that a model, the computer code which pertains to a specific real-world system, is separated from the simulator, the computer program which advances time.

Although there are numerous applications of DEVS to artificial systems, environmental systems, and both biological and physical processes, the formalism is rarely applied to the physics of motion in two or three spatial dimensions. Simulations involving moving gases, liquids, rigid bodies, deformable solids, or a mixture of substances are usually implemented using traditional simulation techniques. Moreover, the timestep is generally fixed, and there is typically no separation between model and simulator. The primary rationale for DEVS-based models of 2D and 3D solids and fluids in motion is the ease with which DEVS-based models can be coupled with one another. For example, researchers and engineers in the field of biomechanics could simulate an implanted medical device by coupling a DEVS model of the device with another DEVS model of the surrounding tissue. Another benefit of DEVS is the formalism's support of different time advancement patterns. For example, the medical device model might be based on an event-driven approach, whereas the tissue model might use either fixed time steps or time steps that shorten in response to fast motion.

Here we apply DEVS to Computational Fluid Dynamics (CFD), the numerical methods and algorithms which solve and analyze the movement and interactions of fluid flows [3]. In general, no analytical solution exists for non-linear fluid models; hence, the numerical approximation methods, also called "computational models," become important. One promising theoretical approach toward resolving CFD-specific problems is the adoption of discrete-event methodologies. CFD Solvers are required to process problems comprised of a large number of computations, which makes the use of computer-based approaches inevitable. In computerized processing of CFD, a boundary for the problem is defined, and the environment is divided into a cellular space in which each cell represents a physical volume. Motion within the defined environment evolves in accordance with the fundamental principles of mass, momentum, and energy conservation. The behavior of the fluid at the boundaries is also defined, termed the boundary

* Corresponding author. Tel.: +1 6135202600.
E-mail addresses: mvschynd@connect.carleton.ca (M. Van Schyndel), gwainer@sce.carleton.ca (G.A. Wainer), rhys.goldstein@autodesk.com (R. Goldstein), jeremy.mogk@autodesk.com (J.P.M. Mogk), azam.khan@autodesk.com (A. Khan).

http://dx.doi.org/10.1016/j.jocs.2014.06.001
1877-7503/© 2014 Elsevier B.V. All rights reserved.

conditions. These specifications construct a model of the fluid which can then be simulated on a powerful computing device. The simulation solves the problem by computing the equations of each cell during a specific duration of time. Finally, visualization and analysis of the results can render a meaningful and sensible outcome of the computations.

Different cellular methods have been proposed to solve these problems. In particular, Cellular Automata (CA) theory [4] is a branch of discrete dynamic systems, in which space is represented by a cellular grid, where each cell is a state machine. In CA, the time advances in a discrete manner and triggers state changes in the cells based on the value of their neighbor cells. This theory has been used in physics, complexity science, theoretical biology, microstructure modeling, and spatial modeling. The Cell-DEVS (Cellular Discrete Event System Specification) formalism [1,5] is a related formalism in which each cell evolves asynchronously using explicit timing delays. This solves the problem of unnecessary processing burden in cells and allows for more efficient asynchronous execution using a continuous time-base, without losing accuracy. In this methodology, each cell is represented as a DEVS atomic model that changes state in response to the occurrence of events in an event-driven fashion.

Cell-DEVS was originally introduced for modeling and simulation of spatial systems; however, not until the current research has anyone proposed using the Cell-DEVS methodology to implement physics-based CFD equations to simulate fluid dynamics. The rule-based nature of defining cellular model behavior provides a platform to define area-wise behavior, leading to easier and faster adoption and implementation of CFD solver algorithms. The other advantage of this method is its fast computing apparatus that works asynchronously on the cellular grid, thus increasing the execution speed. The continuous time-advance nature of Cell-DEVS can contribute to the seamless simulation of CFD, in comparison with the discrete timing in CA that lacks the smoothness of fluid flow. Cell-DEVS models are able to generate realistic results with reasonable speed. Finally, the formal I/O port definitions permit output signals to be produced based on satisfying a specific condition in the cell lattice, and allows the transfer of data between different spatial components. Furthermore, the solver simulation can be interfaced with advanced visualization software to provide a realistic graphical display [6,7].

The solver proposed here uses the DEVS formalism to enable the solution and analysis of fluid flow behaviors using a set of simple and stable CFD algorithms. We describe the Cell-DEVS implementation of these algorithms, supplemented by sample CD++ code, and present simulation results to demonstrate the feasibility of the approach [8]. This is the first successful attempt in modeling CFD as a discrete-event systems specification and we focus on how those results were achieved. Finally, we outline how to address the development of increasingly complex models, including the incorporation of solid barriers and the execution of variable time steps, illustrating its importance within the context of biomechanical and biomedical applications.

2. Related work

Fluid dynamic solvers are used for a wide variety of purposes. Their goal is to create a realistic representation of a naturally occurring fluid system such as rising smoke or blowing dust. The flow of fluids can be viewed as solid particles interacting with velocity fields or as densities. There are different methods for predicting the evolution of these fields and densities: the lattice-gas [9], Navier–Stokes equations [10] and Riemann Solvers [11].

In general, CFD methods are categorized into two groups; (i) Discretization methods and (ii) Turbulence models. Discretization methods are a subset of the divide and conquer approach for solving difficult computational problems, in which the computational domain is discretized and "each term within the partial differential equation describing the flow is written in such a manner that the computer can be programmed to calculate" [12]. Turbulence models are designed to address the unsteady motions that can affect flow, but cannot be directly resolved. The choice of model is typically dictated by the form of the governing equations that were applied, which often relates to the context of the simulation. The model is used to generate solutions at a variety of length and time scales; the more scales that are resolved, the more detailed the flow patterns.

Navier–Stokes equations were the first physical description of fluid motion, a set of differential equations derived from the laws of classical dynamics. The first comprehensive simulation of the Navier–Stokes equations appeared in 1986 [13], and demonstrated that detail to the level of real molecular dynamics was not necessary to cause realistic fluid mechanics. In the book by Sukop et al. [14], a method for creating a basic model of 2D fluid flow is provided, which maps the possible collisions that can occur and the outcomes that are determined by a set procedure. It is the randomness generated by these procedures that is essential to its ability to simulate flows. This procedure does provide reasonable results; however, with the standard of realism ever-increasing, its ability to provide a realistic model is substantially limited.

A similar model was made to represent the effect of polymer chains on fluid flow [15] where a lattice-gas CA was used to provide a 2-dimensional model. It was noted that further work must be done to develop a method of using the lattice-gas method to provide a 3-dimensional model that was able to provide realistic results with a reasonable computational effort.

In a paper by Koelman and Nepveu [16] they demonstrated how it is possible to use a CA to model flow through a porous material. They were able to model a one-phase Darcy automaton based on a Navier–Stokes automaton; however, when they implemented a two-phase Darcy automaton they had to implement much simpler local transition rules. In research presented by Stam [17], the Navier–Stokes equations are used to model the fluid dynamics. While the algorithms implemented do not meet the formalism of CA, they do share several key characteristics. A cell lattice is spanned over the simulation window with each cell holding unique information regarding that particular area. The first difference is that each cell space stores a density value and the horizontal and vertical components of velocity (as well as the z component for a 3-dimensional model). The cell spaces are updated simultaneously at discrete time intervals. In a true CA, each cell can be updated independent of other cells, and the algorithms must solve multiple steps for all cells before the final value is obtained. Nevertheless, Stam's [17] algorithm provided very realistic results with limited computational effort by utilizing a rather basic set of rules, and has potential to be adapted to Cell-DEVS.

In this paper, we use the algorithms presented by Stam to create a CFD solver developed according to the conventions of the Cell-DEVS formalism. These particular algorithms were chosen for several reasons. First, the inherent mathematical stability of these algorithms allows simulations to be advanced using arbitrary time steps. This feature is particularly relevant to the time advancement strategies facilitated by the DEVS formalism. Second, the relative simplicity of these algorithms lends itself well to the prospect of extending this solver to handle increasingly complex scenarios. Third, these algorithms can be performed using a standard PC for reasonably sized grids of both two- and three-dimensions. Fourth, the complete C-code implementation of these algorithms is published in [17], enabling verification of the Cell-DEVS implementation.

Download English Version:

https://daneshyari.com/en/article/429525

Download Persian Version:

https://daneshyari.com/article/429525

Daneshyari.com