Computers & Geosciences 90 (2016) 162–171

Contents lists available at ScienceDirect

Computers & Geosciences

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

Exploring temporal and functional synchronization in integrating models: A sensitivity analysis

Getachew F. Belete *, Alexey Voinov

University of Twente, Department of Geo-Information Processing, PO Box 217, 7500 AE Enschede, Netherlands

ARTICLE INFO

Article history:
Received 20 January 2015
Received in revised form
3 September 2015
Accepted 8 September 2015
Available online 25 September 2015

Keywords:
Time step
Integration method
Component
Web service
Predator-prey
Sensitivity analysis

ABSTRACT

When integrating independently built models, we may encounter components that describe the same processes or groups of processes using different assumptions and formalizations. The time stepping in component models can also be very different depending upon the temporal resolution chosen. Even if this time stepping is handled outside of the components (as assumed by good practice of component building) the use of inappropriate temporal synchronization can produce either major run-time redundancy or loss of model accuracy. While components may need to be run asynchronously, finding the right times for them to communicate and exchange information becomes a challenge. We are illustrating this by experimenting with a couple of simple component models connected by means of Web services to explore how the timing of their input–output data exchange affects the performance of the overall integrated model. We have also considered how to best communicate information between components that use a different formalism for the same processes. Currently there are no generic recommendations for component synchronization but including sensitivity analysis for temporal and functional synchronization should be recommended as an essential part of integrated modeling.

1. Introduction

In integrated modeling we may need to link component models, which are built under different disciplinary paradigms and assumptions, use different temporal and spatial scales, as well as different numeric schemes and methods (Laniak et al., 2013; Peckham et al., 2013). The fact that we are linking and synchronizing potentially very different components, designed to be treated under different spatio-temporal settings, may only add to the uncertainty and variability that can emerge from the integration process itself. The way space and time are treated can be further complicated by the different numeric methods used in components. Using higher order numeric approximations may compensate for some coarser time and space stepping, but may make it more difficult to define appropriate synchronization times and boundaries. Furthermore, components may assume different functional responses when modeling the same processes. Within certain domains these functions may be producing quite similar output, however eventually they can diverge quite significantly only adding to the overall uncertainty of the integration effort.

The investigation of this uncertainty and its impacts on model results can be handled using a kind of sensitivity analysis (Wainwright et al., 2014). In most of the traditional sensitivity studies the focus is on model parameters, including initial conditions (Hamby, 1995). In this research we are specifically looking at sensitivity to model characteristics that are related to integration, to module coupling procedures. As such we will be analyzing the sensitivity of the integrated model to:

1. Variations in time stepping in components and the timing of their synchronization;
2. Changes in numerical methods used in components;
3. Changes in functional responses assumed in components to describe the same processes.

The experiments are conducted by varying only one characteristic at a time and keeping all other controls the same. The observations reported and conclusions drawn from this research can serve as a starting point to perform further sensitivity analysis in integrating models. Our analysis is largely for demonstration purposes to show what we should expect from model coupling and what are the possible problems that we may run into. We have used a very simple, classical model, which we split into components to see how the output will change depending on how the components are run. While there are some good methods for parametric sensitivity analysis, including global sensitivity treatment (Saltelli et al., 2008), these methods hardly apply in our case

* Corresponding author.
E-mail addresses: getfeleke@gmail.com (G.F. Belete), aavoinov@gmail.com (A. Voinov).

http://dx.doi.org/10.1016/j.cageo.2015.09.006
0098-3004/

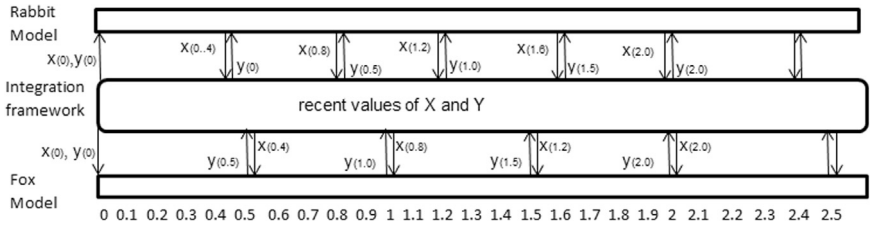

Fig. 1. Data exchange pattern between component models. Rabbit model runs with time step 0.4 and fox model runs with time step 0.5.

when testing sensitivity to how components are organized and coupled. Therefore we had to resort to the trial-and-error type of analysis, simply running the model under different arrangements and reporting the differences observed.

The other reason for doing this analysis is because when modules are linked using data or message exchange approaches, there is always overhead involved. Results from one module have to be collected, packed, sent to another module, unpacked and included in further calculations. This takes time. For example, in the analyses that we present below, the split version of the model runs 13 times slower than when the model is treated as a whole. Clearly we want, when possible, to minimize the interaction between components. When doing that, we want to know what can be gained and what can be lost in terms of accuracy vs. performance.

The paper is organized as follows. Section 2 provides description of the models and integration framework used to perform the sensitivity analysis experiments. In Section 3 three categories of sensitivity analysis experiments with the corresponding observations are presented. Section 4 presents discussion followed by conclusions based on the experiments and observations.

2. The models and the integration framework

Two individual component models and a model integration framework are used to perform this research. The two components are developed based on the classic Lotka–Volterra predator-prey model (Volterra, 1926; Lotka, 1956; Voinov, 2008). The original model mathematically is expressed as:

$$dX/dt = aX - V(X)Y \quad (1)$$

$$dY/dt = cV(X)Y - dY \quad (2)$$

where X=size (or total biomass) of the prey population; Y=size (or total biomass) of the predator population; a=birth rate or number of offspring per individual per year; $V(X)$=so called trophic function that describes the hunting strategy of the prey; c=economic coefficient or efficiency of conversion of prey consumed into new predators; d=mortality rate or proportion of predator population dying per year. In the simplest case $V(X)=bx$, where b=proportion of the prey population consumed by one predator per year.

To convert this model into an integrated, coupled one, we implemented Eq. (1) as an independent rabbit model and Eq. (2) as the fox model. In the first model Y is assumed constant and enters as a parameter, in the second model, similarly, X is constant and is a parameter. When the two models are run in concert they periodically exchange information about X and Y using the most recent value of the variable that is calculated in one model and substituting it for the parameter in the other model. For example, as shown in Fig. 1, if the rabbit model runs with time step 0.4 and fox model runs with time step 0.5, then whenever the time is a multiple of 0.4 the rabbit model will receive the last calculated value of Y from the fox model, and, similarly whenever the step is a multiple of 0.5 the fox model will get the latest reported values of X. Recent values of X and Y are maintained by the model integration framework. When these two components are run with the same time steps, and variables are updated on every time step of the model run, the results are the same as in the original two-variable Lotka–Volterra model solved simultaneously as a system of ordinary differential equations.

The rabbit population dynamics model was built using C++ and the fox model was programmed using Java. Both models are wrapped using Web services so as to enable message-based communication between them (Fig. 2). The web-based model integration framework is built to capture model inputs, to facilitate the communication between them, to manage time steps used, to manage integration types used, and to display the results. Whenever computation by the two models is needed, the input data has to traverse from the integration framework to the Web service wrappers, then to the C++ and Java based implementation of the models, then finally back to the integration framework. The integration framework described above is available at https://github.com/getachewf/mdmif.

As mentioned above, our goal is to study the possible effects of asynchronous and mismatched coupling in a qualitative way, to see what can be potentially expected. In real-life models, which will be certainly of much higher levels of complexity than our simple model, we may be observing other types of behavior. However, even with this simple analysis we can observe some features that are worth mentioning and worth being aware of when coupling model components.

3. Temporal and functional sensitivity analysis in integrating models

To conduct a simulation we have to set parameter values for the model Eqs. (1) and (2) described in the previous section. in setting the parameters we have adopted the parameter values used in the Simile® documentation,[1] and have chosen:

- birth rate for prey, a=0.5,
- proportion of the prey population consumed by one predator per year, b=0.01,
- conversion coefficient of one prey consumed into new predators, c=0.01, i.e. 100 units of rabbit biomass consumed produces one unit of fox biomass,
- mortality rate for predator, d=0.02.

Additionally, for most of simulation runs, we have chosen the following initial values: X_0=5,000 and Y_0=45.

In performing the sensitivity analysis we followed the simple trial-and-error approach. Our sensitivity experiments were mainly grouped into three sets: (1) classic model, same integration and functional schemes in both components, (2) classic model, but

[1] http://www.simulistics.com/.