Simulation of Parallel Agent-based Model of Opinion Exchange within a Shared-memory Multi-core Computing Environment using High Performance Computing

Zhaoya Gong

University of North Carolina at Charlotte

1. Introduction

Agent-based models of complex geographic systems support the representation of interactions among individuals and between individuals and their environments as a means to capture space-time complexity in geographic systems, such as real estate markets or urban land use systems. Without adequate computing support, the representation of agent-based interactions is often over-simplified and abstract, which limits our understanding of the complex nature of geographic systems. Parallel computing technologies utilize advanced computing resources to support the construction of agent-based models that are computationally intensive. As multi-core computing architectures in high-performance and desktop environments become the norm, they provide an unprecedented opportunity for the development of parallelized agent-based models.

2. Objectives

In this paper, we present a parallel computing approach for the development of computationally intensive agent-based modeling within a multi-core computing environment using the High Performance Computing and report on simulation results. This work focuses on the development of a shared-memory parallel agent-based model of opinion exchange and the implementation of the model on a dedicated high performance computer cluster environment. Through the proper utilization of high performance computing resources, simulation could be accomplished with a reasonable timely cost and more experiments regarding to characteristics of agent-agent interactions could be conducted to examine the properties of problem domain.

3. Methodology

We use an agent-based model of opinion exchange simulating the spatial diffusion of individual-level opinion with a focus on agent-agent interactions to support this parallelization effort. Agents in this model are situated within their spatially explicit environments and communicate with each other for opinion development (or knowledge creation). The spatial environment is abstractly represented as a two-dimensional regular grid landscape where each cell associated with an agent. The spatiotemporal dynamics of opinion exchange are driven by the individual-level behavior of agents in a two-step process: every agent first identifies a neighbor agent within certain spatial range, and then communicates with this neighbor agent to exchange opinion.

Our parallel computing approach for this agent-based model is built on a coarse-grained shared memory mechanism to best utilize multi-core computing architectures. We focus our discussions on the parallelization of the agent-based model of opinion exchange by taking advantage of the parallel programming interface OpenMP (Open Multi-Processing) specially designed for utilizing multi-core shared memory computing resources. Two critical issues are identified for the parallelization of the agent-based model: domain decomposition and mutual exclusion. Following the divide-and-conquer principle, the problem domain of opinion exchange between agents is decomposed and assigned to, and coordinated among, multiple computing elements so as to reduce the related computation intensity. During the concurrent processing of each sub-problem domain, mutual exclusion algorithms are appropriately designed to avoid the simultaneous use of a common resource, especially in such a situation that an agent is communicating at the same time with multiple other agents across boundaries of different sub-problem domain handled by separate computing elements. This method insures the consistent update of each agent's opinion during communication but has the potential of slowing down the speedup of parallelization.

4. Implementation

4.1 Spatial neighborhood search

During each iteration agents conduct random searches for other agents that exist within their interaction or communication ranges. Different types of agents may have different communication ranges. The probability, p_{ij}, that agent j is chosen to interact with the current agent i, is determined by a distance-decayed function (see equation 1):

$$p_{ij} = d_{i,j}^{-1/\alpha} \quad (1)$$

where α is a coefficient that represents the influence of distance on the probability that a neighboring agent is chosen–i.e., α determines the rate at which agent interaction becomes less likely as a function of distance. The higher the value of α, the longer the communication range of agent i. $d_{i,j}$ is the distance between two agents. In our model, we only consider those neighborhood cells where the probability to be searched by an agent is larger than a user-defined threshold level p_{thresh} (p_{thresh}=0.01 in this paper). Thus based on equation 1, the communication range of an agent, d_{max}, can be derived using equation 2.

$$d_{\max} = p_{thresh}{}^{-\alpha} \quad (2)$$

4.2 Opinion exchange modeling

Opinion exchange requires agent-agent interactions. We used a bounded confidence model to simulate the opinion exchange process once a neighboring agent is chosen. The opinion of an individual agent is assumed to be a continuous variable normalized on the range of [0, 1]. Each agent possesses two parameters that characterize their opinion exchange behavior: opinion threshold and exchange ratio. If the opinion difference between two agents is less than the opinion threshold of an agent, the opinion of the agent will be updated (see equation 3 and 4).

$$O(i,t) = O(i,t-1) - u_i * \sigma_i \quad (3)$$

$$\sigma_i = O(i,t-1) - O(j,t-1) \quad (4)$$

where

$O(i,t)$, $O(i,t\text{-}1)$: opinions of an agent i at time t and t-1;

$O(j,t-1)$: opinions of an agent j at time t-1;
u_i: opinion exchange ratio of agent i;
σ_i: opinion difference between two agents with respect to agent i.

The opinion exchange ratio is a discount factor that controls the rate of agent opinion change. Agents with high opinion thresholds have high likelihoods of updating their opinion as a result of interactions with other agents. Likewise, the opinions of agents with higher exchange ratios potentially change more quickly as a result of interaction than those of agents with low ratios. The bounded confidence model provides a means of representing complex opinion exchange dynamics. Local-level interactions among agents for opinion exchange lead to the development of landscape-level opinion consensus patterns over time.

4.3 Shared-memory Parallel Agent-based model using OpenMP on High-performance Cluster Computer

Our agent-based model of opinion exchange is implemented within a shared-memory structure using OpenMP API in C programming language. Basically, our agent-based model is characterized by both spatial and temporal dimensions. In terms of spatial dimension, agents interact with one another on a landscape which is represented by a regular grid (e.g. 1000x1000 grid). Each agent is situated on one grid cell (1,000,000 agents in total). The temporal dimension is abstracted by time step. Within each time step every agent is to complete opinion exchange with another agent at least once.

The parallelization of our model is achieved by taking a spatial domain decomposition approach: the entire problem domain (landscape) is equally divided into parts corresponding to the number of threads/cores to be used and each thread/core will process one sub-problem domain (one part) concurrently. In practice, each CPU core will only have one thread created in order to achieve the best performance. For example, if the landscape size is 1000x1000 and an available machine to run this model has 8 cores, the entire landscape will be equally divided by row into 8 parts. And each core/thread will be assigned one of the 8 parts which is 125x1000. Thus, how many parts to be divided will depend on how many cores are available within one machine. Once partitioning is done it will keep exactly the same during the entire simulation. Because changing number of threads and partition will add more complexity and create extra overhead to slow down the simulation process. Another issue under concern is that the inter-thread communication will occur when one agent communicate with another agent which is accidentally located in a separate partition processed by a different thread. Under this situation, mutual exclusion algorithm is adopted to ensure that one agent only communicates with one other agent at the same time. Due to the parallel processing nature, synchronization is triggered after each time step between all concurrent threads/cores upon finishing their own partitions. Specifically, threads completing earlier will wait for other threads finishing their partitions before proceed to next time step. This ensures that all agents equally communicate with the same frequency. The structure of the model is depicted in **Figure1**.

Due to the existence of significant stochastic elements in the model, one run of the model may contain high level of random noises which may be caused by the random number generators for each thread, the communication of different CPU cores within separate slots, and the interruption created by other threads. To reduce those noises to acceptable level, multiple runs of the model are required and the average performance will be calculated based on the results from the multiple runs. Because each run is independent from the others, those model runs could be conducted independently and in parallel. High-performance computing or high throughput parallel computing can play an important role in the implementation of multiple runs of our agent-based model following the principle of embarrassingly parallelism. Each run of our model naturally becomes a job which could be submitted to Open Science Grid (OSG) or TeraGrid by using job scheduling system such as Condor or PBS. This work has been done by using Bash script and PBS syntax to wrap up the model into a PBS job. Specifically, the model written in C is compiled into an executable program and this program is called with its arguments within the job definition script. Another bash script is written to submit this job script in loop (e.g. 100 times) so that our models wrapped in each job are assigned to available computing resources to process in parallel.

Those independent multi-runs of our model also could take advantage of the high throughput nature of OSG. As far as I know, currently eight-way multicore machines are prevalent in the OSG. By defining jobs which designate our model program, model arguments, multi-core computing resources, Condor system can allocate corresponding resources to our models by matching the computing requirements. For example, in Condor job defining file, machines with 4 and plus cores are required and the whole machine is claimed instead of one slot. Ideally, it would be the best for each job to have the same computing configuration and partition pattern such as all using 8-core machines and 8 partitions. But this limits the resources which could be obtained through OSG. Thus, based on my current knowledge it is acceptable to use heterogeneous configurations such as some job running on 4-core machines and others running 8-core machines. However, this direction is still under exploration.

5. Discussion

This is still a going on work. We designed two experiments to examine the improvement of the parallelization of our agent-based model. First experiment is to look at the effect of problem size on the speedup of our parallel agent-based model by changing the landscape size in spatial dimension and the number of computing cores. And the second experiment is to study mutual exclusion on the speedup of parallelization of agent-agent interactions by varying the number of computing threads and the neighborhood size. The computing performance of sequential and parallel versions of our agent-based model is evaluated to demonstrate the utility of shared memory parallel computing approaches in the simulation of complex geographic systems.

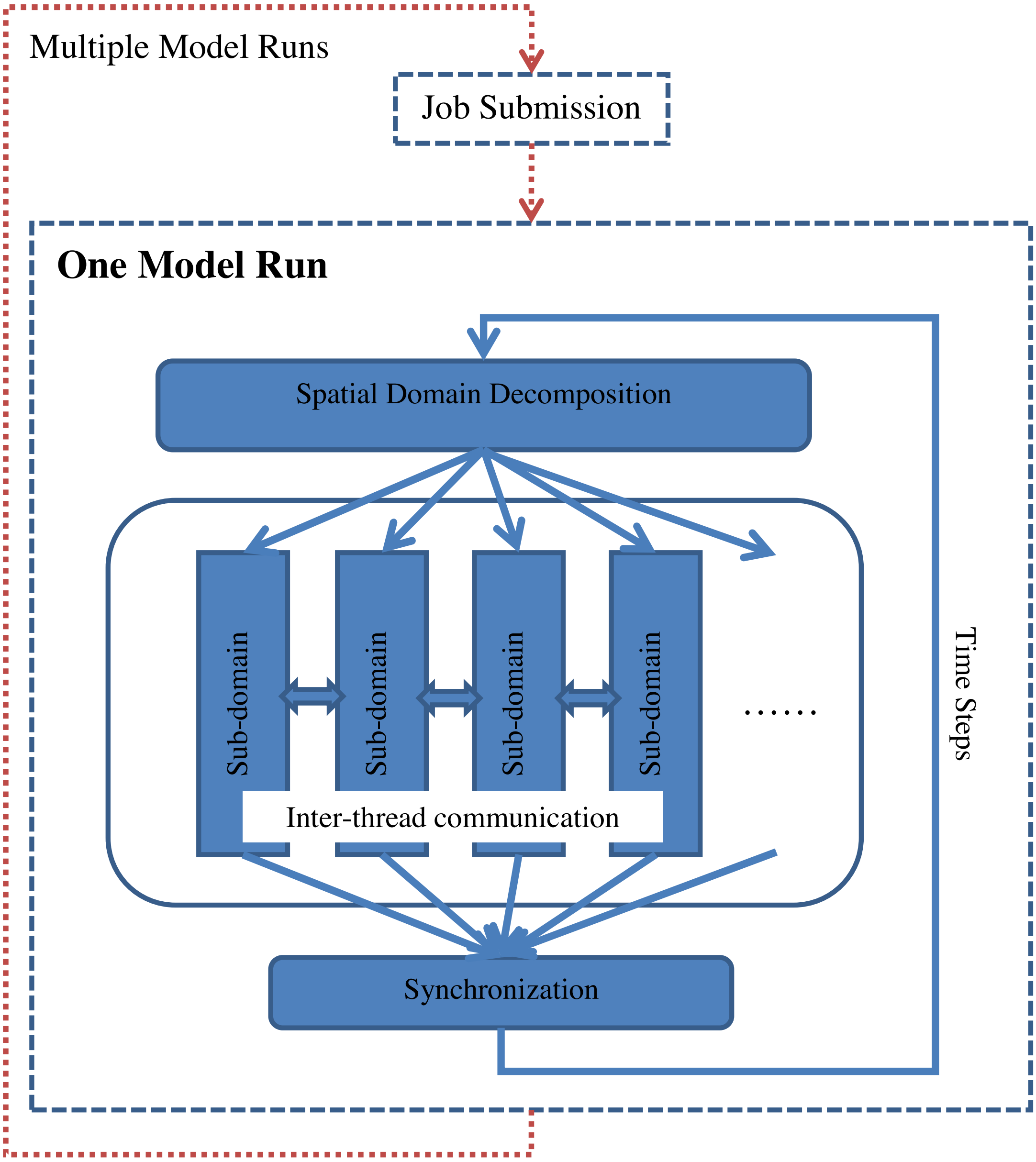

Figure 1 Structure of the model

Reference:

Tang, W. and Bennett, D.A., in revision, Parallel agent-based modeling of spatial opinion diffusion accelerated using Graphics Processing Units, submitted to Ecological Modeling.