Parallel Computing for Geocomputational Modeling

Wenwu Tang, Wenpeng Feng, Jing Deng, Meijuan Jia, and Huifang Zuo

Introduction

In this study, we present the utilization of parallel computing capabilities for geocomputational modeling. Since its emergence in 1990s, geocomputation has been playing a critical role in bridging computer science and geography [1–3]. Geocomputation, as Gahegan [4] identified, is based on four themes in computer science to support geographic problem-solving: (1) *computer architecture and design*, (2) *search, classification, prediction and modeling*, (3) *knowledge discovery*, and (4) *visualization*. Computer algorithms and technologies from these themes are often intertwined to enable the resolution of complex geographic problems through geocomputational modeling. The advancement of these algorithms and technologies in computer science has pushed the development of geocomputation domains. However, gaps often exist between the development of algorithms and technologies in computer science and their applications in geography [1, 5]. Thus, it is necessary to retrospect the development of geocomputational modeling enabled by parallel

W. Tang (✉) • W. Feng • J. Deng • M. Jia
Department of Geography and Earth Sciences, University of North Carolina at Charlotte, Charlotte, NC 28223, USA

Center for Applied GIScience, University of North Carolina at Charlotte, Charlotte, NC 28223, USA
e-mail: WenwuTang@uncc.edu

H. Zuo
Department of Geography and Earth Sciences, University of North Carolina at Charlotte, Charlotte, NC 28223, USA

Center for Applied GIScience, University of North Carolina at Charlotte, Charlotte, NC 28223, USA

Department of Educational Leadership, University of North Carolina, Charlotte, NC 28223, USA

J.-C. Thill, S. Dragicevic (eds.), *GeoComputational Analysis and Modeling of Regional Systems*, Advances in Geographic Information Science,
DOI 10.1007/978-3-319-59511-5_4

computing. The focus of this study is on parallel computing, representative of computer architecture and design in geocomputation themes.

Recent increasing parallel computing applications are attributed to the blossoming of high-performance computing resources in cyberinfrastructure. Cyberinfrastructure, also referred to as e-Science, is the integration of "computing systems, data, information resources, networking, digitally enabled-sensors, instruments, virtual organizations, and observatories, along with an interoperable suite of software services and tools." ([6]; page 1). Cyberinfrastructure, as highlighted by NSF [6], consists of three key capabilities: high-performance and parallel computing, massive data handling and visualization, and virtual organization. High-performance and parallel computing is the key component of cyberinfrastructure that provides massive and transformative computing power for scientific discovery across alternative domains. Domain-specific problems that are infeasible for desktop computing can be solved by using tremendous computing power from cyberinfrastructure-enabled high-performance computing resources [7]. The use of cyberinfrastructure for enhancing problem-solving requires knowledge and skills from computer hardware, software, and specific science domains to best exploit the capabilities of cyberinfrastructure [6–9]. Of course, as cyberinfrastructure continues to develop, requirements on computer knowledge and skills tend to be relaxed. This will greatly urge the domain applications of cyberinfrastructure-enabled high-performance computing. There are a suite of representative cyberinfrastructure, including U.S. XSEDE (Extreme Science and Engineering Discovery Environment; see http://www.xsede.org), Open Science Grid (see http://opensciencegrid.org/), and DEISA (Distributed European Infrastructure for Supercomputing Applications; see http://deisa.eu). High-performance computing resources from these cyberinfrastructures are open to domain scientists for computationally intensive analysis and modeling.

The objective of this chapter is to discuss geocomputational modeling driven by parallel computing at the era of cyberinfrastructure. We organize the remainder of this paper as follows. First, we give an introduction to parallel computing. Then, we provide a detailed discussion on the applications of parallel computing on geocomputational modeling. We focus geocomputational modeling on four aspects: spatial statistics, spatial optimization, spatial simulation as well as cartography and geovisualization. We then use a case study to demonstrate the power of parallel computing for enabling a spatial agent-based model that is computationally challenging. Last, we conclude this chapter and propose directions for future research.

Parallel Computing

Current mainstream computing paradigm is dominated by multi-core and many-core computing, both of which are inherently associated with parallel computing architectures and technologies [10–12]. Multi-core machines are shared-memory computers based on CPU technology, which can be interconnected to form computer

clusters (i.e., distributed memory architectures; see [10, 12]). Many-core computing is fueled by the emergence of NVIDIA many-core GPUs (Graphics Processing Units) for general-purpose computation [13, 14]. Multi- and many-core computing resources are often coupled together—i.e., heterogeneous high-performance computing resources—for the need of parallel computing. These parallel computing architectures serve the basis for cutting-edge cyberinfrastructure-enabled computing, for example, cluster-, Grid-, and cloud-computing (see [10, 15, 16]). In particular, high-performance computing resources are increasingly available on cloud computing platforms [15, 17]. Thus, how to effectively utilize these high-performance computing resources is of greater interest than their accessibility. The solution lies in parallel computing.

Depending on the way that data or information is communicated among processors, two generic types of parallel computing methods exist: message passing and shared memory (see [12]). In message-passing parallel computing, a processor communicates with others for the data required for its subsequent computation through sending and receiving messages. The requested data are encoded into messages on the sender side and then decoded on the receiver side. In terms of shared-memory parallel computing, processors use common address space to exchange data among themselves. Message-passing and shared-memory parallelisms dominate the parallel computing paradigm with a focus on inter-processor communication, which may induce significant overhead. Further, because of inter-processor communication, synchronization is often needed to coordinate concurrent operations among processors. A set of synchronization approaches exists, including barrier, lock, or semaphore [12]. On the other hand, there exist problems for which divided sub-problems do not exchange data or the exchange is light-weighted. In other words, processors will not (frequently) communicate for data from others. For this case, an embarrassingly parallel computing approach (also often referred to as a master-worker approach; see [12]) is the idealistic parallel solution. Because no or little communication among processors exists, high performance on computation is likely to be obtained.

Besides synchronization, a set of parallel strategies, represented by decomposition and load balancing, is often needed to efficiently parallelize a problem. Decomposition strategies support the partitioning of a problem into sub-problems according to characteristics of data (domain decomposition) or functions (functional decomposition) involved [10, 12]. Depending on the size of the sub-problems being partitioned compared with the original problem, decomposition can be fine- or coarse-grained. For spatial problems, spatial domain decomposition that takes into account spatial characteristics of the problems is often used for partitioning and alternative decomposition strategies reported (see [18, 19]).

In particular, Ding and Densham [18] presented detailed discussion on spatial domain decomposition strategies based on the regularity and heterogeneity of spatial domains. As a result, four types (regular versus irregular; homogeneous versus heterogeneous) of spatial domains exist to guide the decomposition. Regarding consideration of interactions or influence among spatial features, Ding and Densham [18] discussed a suite of parallel spatial modeling: local, neighborhood, region,

and global (also see [20]). Ding and Densham [18] suggested that the consideration of spatial characteristics, represented by heterogeneity and dependency, is instrumental in the development of parallel algorithms for spatial problems. For example, spatially heterogeneous characteristics may create an unbalanced distribution of computation across spatial domains of a problem, which will require (more) sophisticated domain decomposition for effectively parallelizing the spatial algorithm. Spatial dependency may affect the choice of the size of neighboring regions, exerting a significant impact on the synchronization mechanism for a spatial problem parallelized using either a message-passing or shared-memory approach.

Once a problem is decomposed into a set of sub-problems, each sub-problem will be wrapped into a task assigned to an individual computing processor. The relationship between tasks and computing processors can be one-to-one or many-to-one. The workload assigned to computing processors may be unbalanced—i.e., load balancing (see [12]) is needed to efficiently utilize the parallel computing resources. Static and dynamic strategies [12] can be applied to achieve load balancing. For static load balancing, tasks are assigned to processors before parallel computing. Once the tasks are executed, there will not be re-assignment of tasks. Optimization algorithms can be used for static load balancing as it is naturally an assignment problem. Dynamic load balancing allows for flexibly reassigning or scheduling tasks among processors to achieve possibly more balanced workload.

To evaluate the performance of parallel algorithms, quantitative metrics based on computing time can be used. Performance metrics mainly include speedup, efficiency, and communication-computation ratio (see [12]). Speedup and efficiency are based on the comparison of execution time between a single processor and multiple processors (Eqs. (1) and (2)). Both speedup and efficiency are positively related to the computing performance of parallel algorithms. Communication-computation ratio is calculated as the ratio of communication time over computation time (Eq. (3)). Heavy communication overhead of a parallel algorithm usually leads to a high communication-computation ratio.

$$s = T_1/T_m \tag{1}$$

$$e = s/n_p \tag{2}$$

$$c = T_{comm}/T_{comp} \tag{3}$$

where s, e, and c are speedup, efficiency, and communication-computation ratio of a parallel algorithm. T_1 denotes the execution time of the sequential algorithm (i.e., on a single processor). T_m is the execution time of the parallel algorithm. T_{comm} is the time spent on inter-processor communication. T_{comp} denotes the time on computation.

Parallel Computing for Geocomputational Modeling

Geocomputational modeling serves as an abstraction of real-world geographic problems. Spatial statistics, spatial optimization, and spatial simulation are three pillars of geocomputational modeling that provide inductive or deductive problem-solving support. Further, geocomputational modeling is inherently related to cartography and geovisualization because of the need of visual presentation of relevant data that are geographically referenced. Thus, in this study, we focus our discussion in terms of the use of parallel computing for geocomputational modeling on four categories: spatial statistics, spatial optimization, spatial simulation, and cartography and geovisualization (Fig. 1). We use articles summarized in Table 1 to guide our discussion.

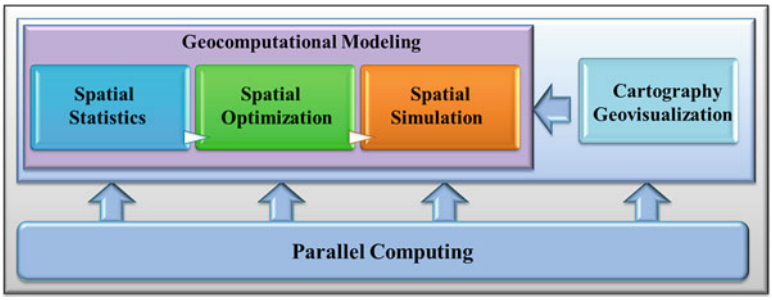

Fig. 1 Illustration of the use of parallel computing for geocomputational modeling

Table 1 List of literature of geocomputational modeling driven by parallel computing

Category	Citation
Spatial statistics	Armstrong et al. [21], Armstrong and Marciano [22], Cheng [23], Gajraj et al. [24], Guan et al. [25], Kerry and Hawick [26], Pesquer et al. [27], Rokos and Armstrong [28], Tang et al. [29] Wang and Armstrong [30], Widener et al. [31], Yan et al. [32]
Spatial optimization	D'Ambrosio et al. [33], Gong et al. [34], He et al. [35], Peredo and Ortiz [36], Porta et al. [37]
Spatial simulation	Abbott et al. [38], Deissenberg et al. [39], Guan and Clarke [40], Li et al. [41], Nagel and Rickert [42], Tang and Wang [43], Tang et al. [44], Tang [45], Uziel and Berry [46], Wang et al. [47]
Cartography geovisualization	Mower [48], Mower [49], Rey et al. [50], Sorokine [51], Tang [52], Vaughan et al. [53], Wang et al. [54], Wang [55]

Spatial Statistics

Spatial statistics provide a means of summarizing spatial characteristics of geographic data or inferring spatial patterns of interest based on first- or second-order properties of these data (see [56, 57]). Spatial statistics mainly comprise spatial autocorrelation analysis (e.g., Moran's *I* or Gerry's *C*), geostatistics (e.g., Kriging interpolation, semivariogram), and spatial pattern analysis (e.g., kernel density analysis, Ripley's *K* approach). Spatial statistics can be univariate, bivariate, or multivariate, thus facilitating the inference of spatial relationships within, between, or among spatial variables [56]. As geographically referenced data are increasingly available with respect to their size and type, spatial statistics provide necessary support for analyzing and understanding spatial characteristics (e.g., heterogeneity and dependence) in these data. Spatial statistics approaches involve comparisons of spatial entities in terms of distance, direction, geometry, or topological relationships [56]. These comparisons may operate at local or global levels with respect to the set of spatial entities [58, 59]. Thus, a significant amount of computation is often required for spatial statistics approaches, particularly when the geographic datasets are large.

Parallel algorithms have been developed for the efficient use of spatial statistics on high-performance computing resources. Armstrong et al. [21] presented their pioneering work in which a $G(d)$ statistic algorithm, functioning as a local spatial cluster approach for hotspot detection [60], was parallelized. Subsequent studies for the parallelization of $G(d)$ algorithm were reported (see [22, 30]). In particular, Wang and Armstrong [30] proposed a formal theory of spatial computational domain and applied it to parallelize the *G* algorithm. Spatial characteristics of geographic data were taken into account in the parallel algorithm to guide the efficient derivation of *G* values. Parallel computing efforts for other spatial statistics algorithms have been reported [28, 29, 31, 32, 61]. For instance, Yan et al. [32] developed a parallel Monte Carlo Markov Chain (MCMC) approach for efficient posterior sampling and applied it to parameterize a Bayesian spatiotemporal model based on Gaussian random field. Widener et al. [31] parallelized the AMOEBA (A Multidirectional Optimal Ecotope-Based Algorithm; see [62]) spatial cluster method using a message-passing approach. The computation of seeds required by the AMOEBA algorithm was partitioned and assigned to individual computing nodes. Tang et al. [29] presented a Ripley's K function approach accelerated through GPUs for spatial point pattern analysis. Acceleration factors, as reported by Tang et al. [29], can reach up to two orders of magnitude on a single Tesla Fermi GPU device and three (about 1501) when using 50 GPUs together.

With respect to geostatistics, Kriging interpolation is an approach that has been actively parallelized in the literature [23–27]. In Guan et al. [25] parallel work, fast Fourier transformation (to derive the covariance matrix) and the computation of weights for Kriging-based areal interpolation were parallelized within a message-passing environment. Guan et al. [25] examined their parallel areal interpolation algorithm on a high-performance computing cluster (about 5000

CPUs) and demonstrated that considerable speed-up was obtained. Pesquer et al. [27] proposed a row-wise decomposition approach to partition the computational load of ordinary Kriging, in which variogram fitting was automated, into a collection of worker nodes. Cheng [23] implemented a GPU-enabled parallel universal Kriging interpolation approach in which computationally intensive matrix-based operations (multiplication) were mapped to many-core architecture on GPUs. As Cheng [23] reported, the acceleration factor by using GPUs for universal Kriging is about 18.

Spatial Optimization

Spatial optimization is to search for optimal solutions from a set of alternatives that constitutes the solution space of a spatial problem of interest [63–65]. A spatial optimization algorithm is converged when its objective function (single- or multi-objective), constrained by a set of criteria, reaches maximum or minimum. Search approaches for optimization algorithms can be exact or heuristic [65]. Exact search enumerates and compares the entire set of solutions, guaranteeing for global optimum. Yet, exact search is only suitable for optimization problems that are relatively simple or small because of the brute-force search of solution space. Heuristic search, including deterministic (e.g., hill-climbing) and stochastic (e.g., simulated annealing, or evolutionary algorithms), introduces automated mechanisms that guide the convergence of the optimization algorithm. While heuristic search does not warrant global optima, it is well-suited to spatial optimization problems that are often sophisticated. Machine learning algorithms (e.g., decision trees, artificial neural networks, evolutionary algorithms, ant colony algorithm, and particle swarm algorithms; see [66–69]) have been extensively used to support heuristics search in optimization algorithms. These machine learning algorithms emulate the behavior of human or animals for intelligent problem-solving. The application of spatial optimization in geography, pioneered by Garrison [70], covers a suite of themes, including site search [71], location analysis [72, 73], spatial planning [74, 75], and ecosystem management [76].

The complexity of geographic problems often leads to a large solution space. As a result, computationally intensive search may be needed in order to obtain (near) optimal solutions for geographic problems, demonstrating the need of parallel computing for spatial optimization. Alternative parallel spatial optimization algorithms have been reported. Peredo and Ortiz [36] developed a simulated annealing algorithm parallelized using a message-passing mechanism to search for spatial patterns that match targeted ones. A tree-based strategy was used to accelerate the computation associated with the acceptance and rejection of perturbed spatial patterns. Machine learning algorithms, for example, artificial neural networks and evolutionary algorithms, have been parallelized. Gong et al. [34] proposed a hybrid parallel neural network algorithm as a nonlinear regression approach for empirical land use modeling. Parallel strategies were applied for the training and validation of ensemble neural networks. For evolutionary algorithms,

the computation of each chromosome is independent with each other. Thus, the population of chromosomes is usually partitioned into a collection of sub-populations each assigned to a computing element for parallel computation. Because of independence among chromosomes' computation, computing performance for parallel evolutionary algorithms is usually high. For example, D'Ambrosio et al. [33] used a parallel evolutionary algorithm for optimal parameter estimation of a debris flow model based on cellular automata, and PGAPack, a parallel evolutionary algorithm software package (see https://code.google.com/p/pgapack/), supported their work. He et al. [35] developed a loose coupling strategy that applied parallel evolutionary algorithms to calibrate two hydrological models. Further, Porta et al. [37] implemented parallel evolutionary algorithms for optimal land use allocation within three types of computing environments: multi-core (shared memory), computing clusters (message passing), and hybrid.

Spatial Simulation

Spatial simulation is an approach that explicitly represents and generates the artificial history of a geographic system [77–79]. Components and their interrelationships in geographic systems are abstracted and represented in spatial simulation models. There are three types of generic spatial simulation [78–80]: system models, cellular automata, and agent-based models. System models, with a foundation in general systems theory [81], employ a set of differential equations to represent macro-level relationships among state variables in a system of interest [82]. Because of the complexity of geographic systems, analytic solutions may not be obtained for these differential equations. Differential equations in system models are often solved using a numerical approach. Cellular automata are based on neighborhood interactions and transition rules to represent spatial dynamics in geographic systems [78]. Agent-based models (or individual-based models) rely on the concept of agents that allow for the explicit representation of decision-making processes of spatially aware individuals or their aggregates [83, 84]. Both cellular automata and agent-based models are bottom-up simulation approaches tailored to the representation of decentralized interactions among components in a geographic system. Besides the three types of generic simulation, there are domain-specific simulation models, for example, hydrological models [85], that have been developed for the study of dynamic spatial phenomena. These simulation approaches (generic and domain-specific) have a vast body of literature in terms of exploring the spatiotemporal complexity of geographic systems.

The representational and generative power of spatial simulation models creates high computational demands, which trigger the motivation of utilizing parallel computing. Costanza and Maxwell [86] detailed the development of a parallel system model for the simulation of coastal landscape dynamics using spatially explicit differential equations. Guan and Clarke [40] presented the parallelization of SLEUTH, a cellular automat model of urban growth, and the application of the

parallel model into the simulation of urban development of the conterminous U.S. Alternative spatial domain decomposition strategies were implemented to partition and allocate computational workload into parallel computing architectures. In Li et al. [41] work, a spatial cellular automata-driven urban simulation model was parallelized with support from strategies of ghost zones (for inter-processor communication) and load balancing (by area or workload). Besides cellular automata, parallel agent-based models have received attention from alternative domain scientists [39, 43, 47]. Uziel and Berry [46] presented a parallel individual-based model to simulate the winter migratory behavior of Yellowstone elk. Regular and irregular spatial domain decomposition strategies were used to cope with the irregular shape of the landscape that elk interacted with. Likewise, Abbott et al. [38] implemented a parallel individual-based model of white-tailed deer in which the foraging and movement of deer on their landscape were partitioned and distributed among multiple processors via a message-passing mechanism. Nagel and Rickert [42] proposed a parallel agent-based simulation of traffic in Portland and used a graph partitioning approach to divide the transportation network in the study area for load-balanced parallel computation. Tang et al. [44] applied a message-passing approach to parallelize a land use opinion model on a supercomputer. Further, as the increasing availability and maturity of GPUs technologies, a suite of parallel spatial simulation models accelerated by using the many-core GPU power have been reported (A detailed review is in [45]).

Cartography and Geovisualization

Cartography and geovisualization enable the presentation of 2- or 3-D spatial data through visual forms (e.g., maps or animations). Cartography has a focus on principals and techniques of mapping [87], while geovisualization is extended from cartography with an emphasis on interactive mapping and on-the-fly visualization of spatial information [88]. Map projection, data classification, generalization, and symbolization constitute fundamental components of cartography and geovisualization [87]. The combination of these cartographic components supports the design of alternative types of maps, including choropleth, dasymetric, isopleth, and proportional symbol or dot maps. Cartography and geovisualization pose a computational challenge [88]. For example, Armstrong et al. [89] illustrated the use of genetic algorithms for optimizing class intervals of choropleth maps and underlined that the process of developing optimal data classification requires computationally intensive search.

Each component in cartography and geovisualization could be highly computationally demanding, for which parallel computing provides a potential solution. Parallel algorithms have been developed to accelerate line simplification (see [49, 53]) and label placement of maps [48]. Tang [52] parallelized the construction of circular cartograms on GPUs. To leverage the massive thread mechanism of GPUs, the construction process was divided to a large number of fine-grained

sub-tasks, while synchronization required by iterations of cartogram construction was conducted at a kernel level. Compared with advanced CPUs, the GPU-based parallel cartogram algorithm obtained a speed up of 15–20. In order to accelerate Fisher-Jenks choropleth map classification, Rey et al. [50] examined three different parallel python libraries, PyOpenCL, Multiprocessing, and Parallel Python, on both CPU-based parallel python and GPU-based PyOpenCL. Their results indicated that satisfactory speedup with the parallelization for moderate to large sample sizes can be achieved and performance gains varied according to different parallel libraries. Advance in high-performance computing greatly encourages the study and application of parallel scientific visualization [54]. Visualization software platforms enabled by high-performance and parallel computing, for example, ParaView (http://www.paraview.org/) and VisIt (https://wci.llnl.gov/codes/visit/home.html) are available and hold promise for accelerating the geovisualization of large geographic data. Sorokine [51] presented a parallel geovisualization module that allowed for leveraging high-performance computing resources for rendering graphics in GRASS GIS. A large geo-referenced image was divided into many smaller tiles concurrently rendered by back-end computing clusters. Similarly, in the work by Wang [55], a map tiling strategy was used for parallel visualization of vector- and raster-based GIS data.

Case Study

Agent-Based Spatial Simulation

In this study, we use a parallel agent-based model of spatial opinion to illustrate the importance and power of parallel computing for geocomputational modeling. The agent-based model was developed and parallelized within GPU environment (see [90]) for detail. In this model, geospatial agents situated within their spatially explicit environments develop and exchange opinions with their neighbors. Each iteration, an agent searches stochastically for its neighbors using a distance-decayed probability function (Eq. (4)).

$$p_{ij} = d_{ij}^{-1/\alpha} \tag{4}$$

The probability (p_{ij}) that two agents (i and j) are peered for opinion exchange is dependent on the distance between them (d_{ij}). After determining which neighbor for communication, the agent will exchange opinion with its neighbor, driven by a bounded confidence model that Weisbuch et al. [91] proposed. In this bounded confidence model, the opinion of an individual is a continuous variable with a range of 0–1. In our model, agents' initial opinions are uniformly randomly distributed. In other words, agents are randomly distributed on their opinion space. Each agent updates its opinion using two parameters: opinion threshold and exchange ratio.

Opinion threshold determines whether the agent will conduct opinion exchange with its neighbor. If the opinion distance between the two agents is shorter than the opinion threshold, the agents will use exchange ratio to update their opinions based on the opinion distance between them. Otherwise, no opinion exchange activities will occur if the opinion distance is longer than the threshold.

To enable the opinion modeling at a large spatial scale, the agent-based opinion model was parallelized and accelerated using general-purpose GPUs (see [90]). NVIDIA CUDA (Compute Unified Device Architecture; see [13, 92]) was the computing platform used for this parallel computing effort. GPU-enabled general purpose computing is based on a shared-memory data parallelism with thread technologies. A large number of CUDA threads are available for concurrently executing the assigned computing tasks on the streaming processors of a GPU device. In this study, the population of geospatial agents was divided into a collection of sub-populations based on a 1D block-wise domain decomposition strategy (see [11, 18]). Each sub-population may consist of one or multiple agents and the associated opinion development process is handled by a CUDA thread. Because the number of threads allowed in CUDA-enabled GPUs is large, massive agents are supported in this parallel spatial model.

Experiment

We designed an experiment to examine how parallel computing accelerates and thus facilitates the agent-based modeling of large-scale spatial opinion exchange. The experiment is to investigate the impact of communication range on the spatial opinion exchange. In this model, the distance coefficient (α in Eq. (4)) in the distance-decayed neighborhood search determines the communication range (see [90]). We varied this distance coefficient from 0.2 to 1.3 at an interval of 0.1 (corresponding to 3–398 cells). Consequently, there are 12 treatments in this experiment (noted as T1–T12). The distance threshold of agents was set at 0.22 and the exchange ratio is 0.40. A raster landscape was used in this study, and the landscape size of the model is 2000×2000. Each cell is situated by an agent. For each treatment, we repeated the model run 100 times, in total 1200 runs required. GPU devices that we used in this study are Nvidia Tesla Fermi M2050 (including 448 cores). CPUs are dual Intel Xeon hex-core processors (clock rate: 2.67 GHz; memory: 12 GBs).

In this model, as agents communicate with their neighbors, their opinions tend to move towards each other in their opinion space. When their opinions are clustered within a small range, these agents reach consensus. In this experiment, we are interested in the consensus development of agents. So we used an index of entropy [93] to quantify the spatial opinion patterns over time. The entropy index allows for representing the diversity of spatial opinions: a high entropy index illustrates that the spatial opinion pattern is diverse. Otherwise, a low entropy index is associated

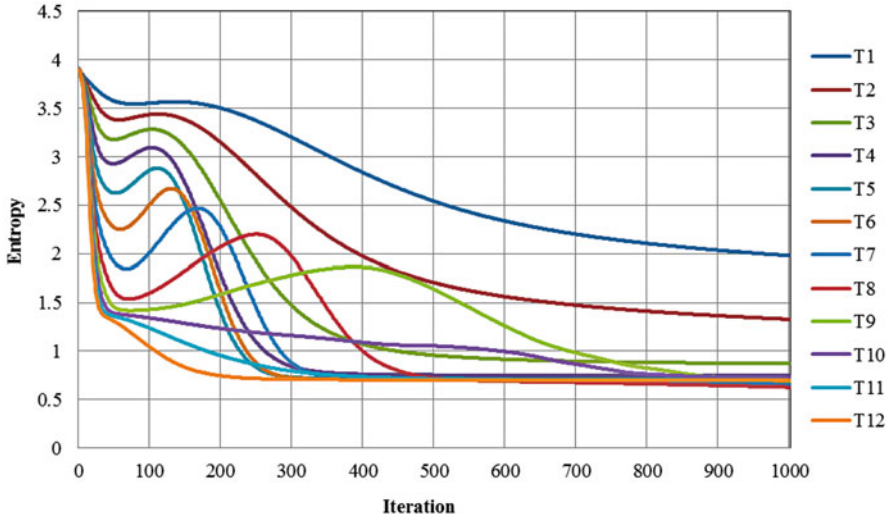

Fig. 2 Time series of opinion entropies over iterations (T1–T12: treatment 1–12)

with a homogeneous spatial opinion pattern—i.e., agent opinions are converged or consensus is reached.

$$en = -\sum_{i=1}^{n} p_i^* \log p_i \tag{5}$$

where en is the entropy of an agent opinion pattern. p_i is the probability of opinion group i. n is the number of opinion groups. Figure 2 shows the time series of Shannon's entropy over 1000 iterations for the 12 treatments. For the first treatment, the communication range is short (threecells). The total number of possible neighbors that an agent exchanges opinion is small (79 neighbors). Thus, as agents communicate for exchanging opinions, entropy exhibits a gradually decreasing pattern. The averaged entropy at iteration 1000 is about 2.0. In other words, agents' opinions do not converge because of the limited communication range.

As increase in communicate range, entropy curves tend to reach minimum quickly. In most of the treatments entropy values converge within a range of 0.5–1.0. This illustrates that increment in communication range tends to increase the likelihood of communicating with more agents with diverse opinions. As a result, it is easier for agent opinion to converge for consensus. Of interest is the pattern of convergence iterations and entropies as communication ranges increase (see Fig. 3). For the first seven treatments, both convergence time and corresponding entropies tend to be lower when communication ranges increase. Yet, once communication range exceeds 40 cells (treatment eight and after), convergence time exhibits a wide range of variation (between iteration 1 and 1000). Most of the model runs

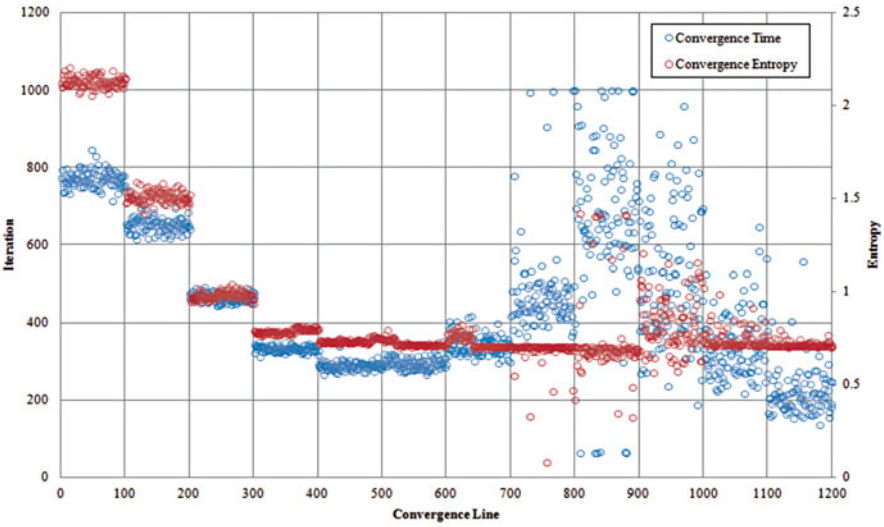

Fig. 3 Convergence iterations and entropies for model runs in the experiment

(except treatment T10) tend to converge at a small range of values for each treatment. Communication range of 40 cells is a critical threshold that triggers the state transition between stable and unstable convergence. This can be attributed to change in the interacting intensity (the amount of agent interactions per unit area) required for spatial opinion exchange. Before communication range reaches 40 cells, interacting intensity (given the same amount and types of agents) is high such that agents have sufficient opportunities to exchange their opinions for consensus. Yet, once the communication range exceeds 40 cells, interacting intensity required for opinion convergence tends to decrease (the number of interactions remains the same, but neighboring zones are enlarged). The decreased interacting intensity produces a form of diluting effect that introduces instability in the convergence of agent opinion.

We used acceleration factor, ratio of computing time on a single CPU over that on a GPU device (similar to speedup; see [90]; cf. [12]), to evaluate the computing performance of the parallel agent-based opinion model. Table 2 reports results of computing performance (including computing time and acceleration factor) of the 12 treatments. GPU computing time of each model run varies between 150 and 160 s (about 2–3 min), and corresponding CPU computing time falls within a range of 1600–1800 s (about half an hour per run). So the computing time required by this experiment is reduced from 600 h for a single CPU (0.5 h per run $\times$ 1200 runs; about 25 days) to 60 h for a single GPU. About 10–12 acceleration factors per GPU device per run were obtained. Because each run in this experiment is independent, we used 30 GPUs to concurrently execute these model runs to achieve further acceleration. The total CPU-based sequential computing time of this experiment requires 23.58 days. When using 30 GPUs together, it takes 6730.11 s to complete

Table 2 Results of computing performance of the agent-based modeling of spatial opinion exchange (time unit: seconds; Std: standard deviation)

	CPU time		GPU time		Acceleration factor	
Treatment	Mean	Std	Mean	Std	Mean	Std
T1	1773.32	142.28	158.93	10.69	11.23	0.89
T2	1685.88	116.88	158.37	10.49	10.84	1.04
T3	1655.66	100.26	157.95	11.99	10.58	1.09
T4	1670.81	117.33	156.04	12.51	10.66	1.12
T5	1652.46	105.76	159.71	9.69	10.24	0.87
T6	1644.99	86.87	158.53	11.02	10.52	1.18
T7	1675.75	120.79	153.85	11.54	11.04	1.27
T8	1659.36	105.97	164.69	3.15	10.11	0.71
T9	1679.23	112.68	156.38	12.56	10.89	1.15
T10	1660.30	105.06	155.97	10.54	10.46	0.94
T11	1677.63	119.56	159.75	10.10	10.39	0.90
T12	1756.70	141.86	160.18	11.12	11.14	1.41

the experiment. The corresponding acceleration factor (similar to speed up) for completing the entire experiment is 302.74 with respect to a single CPU. The influence of communication range on computing performance (both computing time and acceleration factors) is insignificant in this experiment.

Conclusion

In this study, we illustrated the power of parallel computing for geocomputational modeling and identified parallel strategies instrumental in tackling the associated computationally intensive issues. High-performance computing technologies are extensively available for domain-specific scientists in general and geographers in particular. Parallel computing strategies, represented by decomposition, synchronization, and communication, allow for best utilizing parallel computing architectures that high-performance computing is built on. In particular, for the parallelization of spatially explicit geocomputational modeling, spatial characteristics can be taken into account into parallel spatial algorithms to best leverage the high-performance computing capabilities of state-of-the-art cyberinfrastructure.

We focused our discussion on four categories related to geocomputational modeling: spatial statistics, spatial optimization, spatial simulation, and cartography and geovisualization. The first three approaches (statistics, optimization, and simulation) serve as the pillars of geocomputational modeling. These three approaches allow us to abstract and transform geographic problems into geocomputational modeling. The abstraction and representation of these problems in geocomputational modeling approaches make them computationally challenging. Parallel computing provides a potential solution to resolve the computational intensity of these geocomputational

modeling approaches. Cartography and geovisualization support the visual presentation of geo-referenced data or information associated with geocomputational modeling. The use of parallel computing for the acceleration of cartography and geovisualization methods is needed when massive data are associated with, or produced from, geocomputational modeling.

Future research directions that we suggest for the applications of parallel computing in geocomputational modeling include (1) more elegant parallel spatial strategies for the best utilization of computing power in alternative high-performance computing resources, including heterogeneous multi- and many-core computing architecture; (2) more detailed investigation on the capability of parallel geocomputational modeling approaches (statistics, optimization, and simulation) for large-scale spatial problem-solving; and (3) parallel geovisualization technologies for the visual presentation of large GIS data and information (i.e., big data) associated with geocomputational modeling.

References

1. Armstrong MP (2000) Geography and computational science. Ann Assoc Am Geogr 90: 146–156
2. Longley PA (1998) Foundations. In: Longley PA, Brooks SM, McDonnell R, MacMillan B (eds) Geocomputation: a Primer. Wiley, New York
3. Openshaw S, Abrahart RJ (1996) Geocomputation. In: Abrahart RJ (ed) Proceedings of the first international conference on geocomputation. University of Leeds, Leeds, pp 665–666
4. Gahegan M (1999) What is geocomputation? Trans GIS 3:203–206
5. Openshaw S, Turton I (2000) High performance computing and art of parallel programming: an introduction for geographers, social scientists, and engineers. Taylor & Francis Group, London
6. NSF (2007) Cyberinfrastructure vision for 21st century discovery. Report of NSF Council. http://www.nsf.gov/od/oci/ci_v5.pdf
7. Atkins DE, Droegemeie KK, Feldman SI, Garcia-Molina H, Klein ML, Messerschmitt DG et al (2003) Revolutionizing science and engineering through cyberinfrastructure: report of the National Science Foundation Blue-Ribbon Advisory Panel on cyberinfrastructure. US National Science Foundation, Arlington, VA
8. Wang S (2010) A CyberGIS framework for the synthesis of cyberinfrastructure, GIS, and spatial analysis. Ann Assoc Am Geogr 100:535–557
9. Yang C, Raskin R, Goodchild M, Gahegan M (2010) Geospatial cyberinfrastructure: past, present and future. Comput Environ Urban Syst 34:264–277
10. Dongarra J, Foster I, Fox G, Gropp W, Kennedy K, Torczon L et al (eds) (2003) The sourcebook of parallel computing. Morgan Kaufmann, San Francisco, CA
11. Foster I (1995) Designing and building parallel programs: concepts and tools for parallel software engineering. Addison-Wesley, Reading, MA
12. Wilkinson B, Allen M (2004) Parallel programming: techniques and applications using networked workstations and parallel computers, Second edn. Pearson Prentice Hall, Upper Saddle River, NJ
13. Kirk DB, Hwu W-m (2010) Programming massively parallel processors: a hands-on approach. Morgan Kaufmann, Burlington, MA
14. Owens JD, Luebke D, Govindaraju N, Harris M, Krüger J, Lefohn AE et al (2007) A survey of general-purpose computation on graphics hardware. Comput Graph Forum 26:80–113

15. Armbrust M, Fox A, Griffith R, Joseph A, Katz R, Konwinski A et al (2010) A view of cloud computing. Commun ACM 53:50–58
16. Foster I, Kesselman C (eds) (2004) The grid: blueprint for a new computing infrastructure. Morgan Kaufmann, San Francisco, CA
17. Yang C, Goodchild M, Huang Q, Nebert D, Raskin R, Xu Y et al (2011) Spatial cloud computing: how can the geospatial sciences use and help shape cloud computing? Int J Digital Earth 4:305–329
18. Ding YM, Densham PJ (1996) Spatial strategies for parallel spatial modelling. Int J Geogr Inf Syst 10:669–698
19. Wang S, Armstrong MP (2003) A quadtree approach to domain decomposition for spatial interpolation in grid computing environments. Parallel Comput 29:1481–1504
20. Tomlin DC (1990) Geographic information systems and cartographic modeling. Prentice Hall, Englewood Cliffs, NJ
21. Armstrong M, Pavlik C, Marciano R (1994) Parallel processing of spatial statistics. Comput Geosci 20:91–104
22. Armstrong M, Marciano R (1995) Massively parallel processing of spatial statistics. Int J Geogr Inf Syst 9:169–189
23. Cheng T (2013) Accelerating universal Kriging interpolation algorithm using CUDA-enabled GPU. Comput Geosci 54:178–183
24. Gajraj A, Joubert W, Jones J (1997) A parallel implementation of kriging with a trend. Report LA-UR-97-2707. Los Alamos National Laboratory, Los Alamos
25. Guan Q, Kyriakidis P, Goodchild M (2011) A parallel computing approach to fast geostatistical areal interpolation. Int J Geogr Inf Sci 25:1241–1267
26. Kerry KE, Hawick KA (1998) Kriging interpolation on high-performance computers. Technical report DHPC-035. Department of Computer Science, University of Adelaide, Australia
27. Pesquer L, Cortés A, Pons X (2011) Parallel ordinary kriging interpolation incorporating automatic variogram fitting. Comput Geosci 37:464–473
28. Rokos, Armstrong MP (1996) Using Linda to compute spatial autocorrelation in parallel. Comput Geosci 22:425–432
29. Tang W, Feng W, Jia M (2015) Massively parallel spatial point pattern analysis: Ripley's K function accelerated using graphics processing units. Int J Geogr Inf Sci 29:412–439
30. Wang S, Armstrong M (2009) A theoretical approach to the use of cyberinfrastructure in geographical analysis. Int J Geogr Inf Sci 23:169–193
31. Widener M, Crago N, Aldstadt J (2012) Developing a parallel computational implementation of AMOEBA. Int J Geogr Inf Sci 26:1707–1723
32. Yan J, Cowles M, Wang S, Armstrong M (2007) Parallelizing MCMC for Bayesian spatiotemporal geostatistical models. Stat Comput 17:323–335
33. D'Ambrosio D, Spataro W, Iovine G (2006) Parallel genetic algorithms for optimising cellular automata models of natural complex phenomena: an application to debris flows. Comput Simul Nat Phenom Hazard Assess 32:861–875
34. Gong Z, Tang W, Thill J (2012) Parallelization of ensemble neural networks for spatial land-use modeling. In: Proceedings of the 5th international workshop on location-based social networks. ACM, Redondo Beach, CA, pp 48–54
35. He K, Zheng L, Dong S, Tang L, Wu J, Zheng C (2007) PGO: a parallel computing platform for global optimization based on genetic algorithm. Comput Geosci 33:357–366
36. Peredo O, Ortiz J (2011) Parallel implementation of simulated annealing to reproduce multiple-point statistics. Comput Geosci 37:1110–1121
37. Porta J, Parapar J, Doallo R, Rivera F, Santé I, Crecente R (2013) High performance genetic algorithm for land use planning. Comput Environ Urban Syst 37:45–58
38. Abbott CA, Berry MW, Comiskey EJ, Gross LJ, Luh H-K (1997) Parallel individual-based modeling of Everglades deer ecology. Comput Sci Eng IEEE 4:60–78
39. Deissenberg C, van der Hoog S, Dawid H (2008) EURACE: a massively parallel agent-based model of the European economy. Appl Math Comput 204:541–552

40. Guan Q, Clarke K (2010) A general-purpose parallel raster processing programming library test application using a geographic cellular automata model. Int J Geogr Inf Sci 24:695–722
41. Li X, Zhang X, Yeh A, Liu X (2010) Parallel cellular automata for large-scale urban simulation using load-balancing techniques. Int J Geogr Inf Sci 24:803–820
42. Nagel K, Rickert M (2001) Parallel implementation of the TRANSIMS micro-simulation. Parallel Comput 27:1611–1639
43. Tang W, Wang S (2009) HPABM: a hierarchical parallel simulation framework for spatially-explicit agent-based models. Trans GIS 13:315–333
44. Tang W, Bennett D, Wang S (2011) A parallel agent-based model of land use opinions. J Land Use Sci 6:121–135
45. Tang W (2013a) Accelerating agent-based modeling using Graphics Processing Units. In: Shi X, Volodymyr K, Yang C (eds) Modern accelerator technologies for GIScience. Springer, New York, pp 113–129
46. Uziel E, Berry MW (1995) Parallel models of animal migration in Northern Yellowstone National Park. Int J High Perform Comput Appl 9:237–255
47. Wang D, Berry M, Carr E, Gross L (2006) A parallel fish landscape model for ecosystem modeling. Simulation 82:451–465
48. Mower J (1993) Automated feature and name placement on parallel computers. Cartogr Geogr Inf Syst 20:69–82
49. Mower JE (1996) Developing parallel procedures for line simplification. Int J Geogr Inf Syst 10:699–712
50. Rey SJ, Anselin L, Pahle R, Kang X, Stephens P (2013) Parallel optimal choropleth map classification in PySAL. Int J Geogr Inf Sci 27:1023–1039
51. Sorokine A (2007) Implementation of a parallel high-performance visualization technique in GRASS GIS. Comput Geosci 33:685–695
52. Tang W (2013b) Parallel construction of large circular cartograms using graphics processing units. Int J Geogr Inf Sci 27(11):1–25
53. Vaughan J, Whyatt D, Brookes G (1991) A parallel implementation of the Douglas-Peucker line simplification algorithm. Softw Pract Exp 21:331–336
54. Wang L, Chen D, Deng Z, Huang F (2011) Large scale distributed visualization on computational grids: a review. Comput Electr Eng 37:403–416
55. Wang H (2012) A large-scale dynamic vector and raster data visualization geographic information system based on parallel map tiling [Thesis]. Florida International University, Miami, FL
56. Cressie NA (1993) Statistics for spatial data (revised edition). Wiley, New York
57. Ripley BD (2005) Spatial statistics. Wiley, Hoboken
58. Anselin L (1995) Local indicators of spatial association—LISA. Geogr Anal 27:93–115
59. Getis A, Ord JK (1996) Local spatial statistics: an overview. In: Longley PA, Batty M (eds) Spatial analysis: modelling in a GIS environment. Wiley, New York
60. Getis A, Ord JK (1992) The analysis of spatial association by use of distance statistics. Geogr Anal 24:189–206
61. Zhang J (2010) Towards personal high-performance geospatial computing (HPC-G): perspectives and a case study. In: Proceedings of the ACM SIGSPATIAL international workshop on high performance and distributed geographic information systems. ACM, San Jose, CA, pp 3–10
62. Aldstadt J, Getis A (2006) Using AMOEBA to create a spatial weights matrix and identify spatial clusters. Geogr Anal 38:327–343
63. Deb K (2001) Multi-objective optimization. In: Multi-objective optimization using evolutionary algorithms. Wiley, West Sussex, pp 13–46
64. Fletcher R (2013) Practical methods of optimization. Wiley, New York
65. Tong D, Murray AT (2012) Spatial optimization in geography. Ann Assoc Am Geogr 102:1290–1309
66. Bishop CM, Nasrabadi NM (2006) Pattern recognition and machine learning. Springer, New York

67. Bonabeau E, Dorigo M, Theraulaz G (1999) Swarm intelligence: from natural to artificial systems. Oxford University Press, New York
68. Goldberg DE, Holland JH (1988) Genetic algorithms and machine learning. Mach Learn 3: 95–99
69. Russell SJ, Norvig P, Canny JF, Malik JM, Edwards DD (1995) Artificial intelligence: a modern approach. Prentice Hall, Upper Saddle River, NJ
70. Garrison WL (1959) Spatial structure of the economy: II. Ann Assoc Am Geogr 49:471–482
71. Cova TJ, Church RL (2000) Exploratory spatial optimization in site search: a neighborhood operator approach. Comput Environ Urban Syst 24:401–419
72. Church RL (1990) The regionally constrained p-median problem. Geogr Anal 22:22–32
73. Murray AT, Gottsegen JM (1997) The influence of data aggregation on the stability of p-median location model solutions. Geogr Anal 29:200–213
74. Aerts JCJH, Eisinger E, Heuvelink GBM, Stewart TJ (2003) Using linear integer programming for multi-site land-use allocation. Geogr Anal 35:148–169
75. Scott AJ (1971) Combinatorial programming, spatial analysis and planning. Methuen, London
76. Hof JG, Bevers M (1998) Spatial optimization for managed ecosystems. Columbia University Press, New York
77. Banks J (1998) Handbook of simulation. Wiley, New York
78. Batty M (2005) Cities and complexity: understanding cities with cellular automata, agent-based models, and fractals. The MIT Press, Cambridge, MA
79. Benenson I, Torrens PM (2004) Geosimulation: automata-based modeling of urban phenomena. Wiley, London
80. Parker DC, Manson SM, Janssen MA, Hoffmann MJ, Deadman P (2003) Multi-agent systems for the simulation of land-use and land-cover change: a review. Ann Assoc Am Geogr 93: 314–337
81. Von Bertalanffy L (1972) The history and status of general systems theory. Acad Manag J 15:407–426
82. Costanza R, Voinov A (2004) Landscape simulation modeling: a spatially explicit, dynamic approach. Springer, New York
83. Epstein JM (1999) Agent-based computational models and generative social science. Complexity 4:41–60
84. Grimm V, Railsback SF (2005) Individual-based modeling and ecology. Princeton University Press, Princeton, NJ
85. Gassman PW, Reyes MR, Green CH, Arnold JG (2007) The soil and water assessment tool: historical development, applications, and future research directions. Trans Agric Biol Eng 50:1211–1250
86. Costanza R, Maxwell T (1991) Spatial ecosystem modelling using parallel processors. Ecol Model 58:159–183
87. Slocum TA, McMaster RB, Kessler FC, Howard HH (2009) Thematic cartography and geovisualization. Pearson Prentice Hall, Upper Saddle River, NJ
88. MacEachren AM, Gahegan M, Pike W, Brewer I, Cai G, Lengerich E et al (2004) Geovisualization for knowledge construction and decision support. Comput Graph Appl IEEE 24:13–17
89. Armstrong MP, Xiao N, Bennett DA (2003) Using genetic algorithms to create multicriteria class intervals for choropleth maps. Ann Assoc Am Geogr 93(3):595–623
90. Tang W, Bennett DA (2011) Parallel agent-based modeling of spatial opinion diffusion accelerated using graphics processing units. Ecol Model 222:3605–3615
91. Weisbuch G, Deffuant G, Amblard F, Nadal J-P (2002) Meet, discuss, and segregate. Complexity 7:55–63
92. CUDA (2016) CUDA. http://www.nvidia.com/object/cuda_home_new.html
93. Shannon CE (1948) A mathematical theory of communication. Bell Syst Tech J 27:379–423