INTERNATIONAL JOURNAL OF DIGITAL EARTH
2019, VOL. 12, NO. 5, 566–582
https://doi.org/10.1080/17538947.2018.1464073

Parallel computing solutions for Markov chain spatial sequential simulation of categorical fields

Weixing Zhang[a,b,c], Weidong Li[a,b], Chuanrong Zhang[a,b] and Tian Zhao[d]

[a]Department of Geography, University of Connecticut, Storrs, USA; [b]Center for Environmental Science and Engineering, University of Connecticut, Storrs, USA; [c]Connecticut State Data Center, University of Connecticut, Storrs, USA; [d]Department of Computer Science, University of Wisconsin–Milwaukee, Milwaukee, USA

ABSTRACT
The Markov chain random field (MCRF) model is a spatial statistical approach for modeling categorical spatial variables in multiple dimensions. However, this approach tends to be computationally costly when dealing with large data sets because of its sequential simulation processes. Therefore, improving its computational efficiency is necessary in order to run this model on larger sizes of spatial data. In this study, we suggested four parallel computing solutions by using both central processing unit (CPU) and graphics processing unit (GPU) for executing the sequential simulation algorithm of the MCRF model, and compared them with the nonparallel computing solution on computation time spent for a land cover post-classification. The four parallel computing solutions are: (1) multicore processor parallel computing (MP), (2) parallel computing by GPU-accelerated nearest neighbor searching (GNNS), (3) MP with GPU-accelerated nearest neighbor searching (MP-GNNS), and (4) parallel computing by GPU-accelerated approximation and GPU-accelerated nearest neighbor searching (GA-GNNS). Experimental results indicated that all of the four parallel computing solutions are at least 1.8× faster than the nonparallel solution. Particularly, the GA-GNNS solution with 512 threads per block is around 83× faster than the nonparallel solution when conducting a land cover post-classification with a remotely sensed image of 1000 × 1000 pixels.

ARTICLE HISTORY
Received 28 November 2017
Accepted 8 April 2018

KEYWORDS
Markov chain random field; parallel computing; nearest neighbor searching; approximation; graphics processing unit

1. Introduction

Geostatistical (or spatial statistical) simulation algorithms (e.g. sequential simulation, turning bands, truncated Gaussian, and simulated annealing) (Srivastava 1996), particularly sequential simulation, have been widely used in geographic information fields, including uncertainty assessment (Mowrer 1997; Goovaerts 2002; Zhao et al. 2005), soil/sediment mapping (Bierkens and Burrough 1993; Weissmann and Fogg 1999; Lark 2000; Wu et al. 2004; Zhang and Li 2008), environmental evaluation (Lee and Ellis 2000; Juang, Chen, and Lee 2004), land cover post-classification (Zhang, Li, and Zhang 2016; Zhang et al. 2017a) and urban growth detection (Zhang et al. 2017b). However, spatial sequential simulation is computationally costly, particularly when simulating a massive amount of nodes (Vargas, Caetano, and Mata-Lima 2008; Mariethoz 2010; Nunes and Almeida 2010; Peredo, Ortiz, and Herrero 2015; Rasera, Machado, and Costa 2015). As a new geostatistical approach for dealing with categorical spatial variables, the Markov chain random field (MCRF) approach (Li 2007a) adopted the spatial sequential simulation idea in its simulation algorithm

CONTACT Chuanrong Zhang chuanrong.zhang@uconn.edu Department of Geography, University of Connecticut, Storrs, CT 06269-4148, USA; Center for Environmental Science and Engineering, University of Connecticut, Storrs, CT 06269-4148, USA

(see Li and Zhang 2007 for the Markov chain sequential class simulation algorithm). As an extension of the MCRF model, the MCRF cosimulation (coMCRF) model (Li et al. 2015) has been proposed and successfully used in land cover post-classification recently for improving the overall accuracy of land cover classification (Zhang, Li, and Zhang 2016; Zhang et al. 2017a; Zhang et al. 2017b). However, similar to other spatial sequential simulation methods, the computational cost of executing the coMCRF model in sequential simulation increases dramatically as the data size of a simulation increases. Therefore, better computational performance of the coMCRF model would significantly encourage the use of the model in land cover post-classification and other applications of the MCRF approach.

Parallel computing techniques have created a great opportunity for resolving spatial computational issues (Wang and Armstrong 2009; Ingram and Cornford 2010; Chao et al. 2011; Zhao et al. 2015; Li, Hodgson, and Li 2016; Liu et al. 2016; Du et al. 2017). At present, parallel computing solutions can be divided into three groups in terms of central processing units (CPUs) and graphics processing units (GPUs). The first group is parallelization using multi-core CPU(s) (including both shared memory models and distributed memory models). In geostatistics, for example, a number of researchers parallelized kriging interpolation processes on distributed computing systems (Armstrong and Marciano 1997; Pesquer, Cortés, and Pons 2011; Wei et al. 2015). Peredo, Ortiz, and Herrero (2015) enabled the Geostatistical Software Library (GSL) to improve computation efficiency using both hybrid parallel computing and code optimization. Mariethoz (2010) presented a general parallelization strategy for geostatistical sequential simulation at the path level and tested it using Message Passing Interface (MPI) on a distributed computing system. The second group is parallelization using many-core GPU(s) (e.g. the Compute Unified Device Architecture (CUDA) framework). For example, Srinivasan, Duraiswami, and Murtugudde (2010) parallelized the process of kriging interpolation to approximate real-time atmospheric conditions from scattered data on a GPU device. In order to reduce the execution time of running an ordinary kriging model, De Ravé et al. (2014) parallelized the matrix inverse processes, which were considered the most time-consuming step within the model. Li et al. (2014) developed a two-stage GPU-accelerated method to achieve the speedup of 17× in high-order spatial statistics calculation. Mei (2014) further improved the computational efficiency of Inverse Distance Weighting interpolation on GPU by maximizing the use of fast shared memory. Cheng (2013) and Wu et al. (2016) succeeded in speeding up a kriging algorithm for interpolation with the aid of CUDA-enabled GPU. The third group is heterogeneous computing. For example, in order to accelerate geostatistical simulation, Tahmasebi et al. (2012) developed a hybrid parallel computing method that can avoid conflicts between concurrent computations in simulation. Shi and Ye (2013) conducted a comparative study on parallelization methods of kriging interpolation on hybrid computer architectures and showed that the use of multiple GPU devices can accelerate kriging interpolation over a large data set.

However, applying parallel computing to spatial sequential simulation is not easy, mainly because the requirement of parallelization is often against the high dependence between the tasks from sequential simulation (Nunes and Almeida 2010; Tahmasebi et al. 2012). This study explored the utilization of parallel computing techniques on sequential simulation of the MCRF approach. Considering that the recently proposed coMCRF model is very useful in land cover post-classification but it is time-consuming with large data sets, we used the coMCRF model in the MCRF sequential simulation algorithm. Experiments were performed on land cover post-classification. In order to evaluate whether parallel computing solutions can reproduce the results of nonparallel sequential simulation, we conducted a comparison between simulation results of the proposed parallel computing solutions and that of the nonparallel solution. In addition, we also considered different numbers of CPU cores and GPU threads in parallel computing. Although the experiments were made on the coMCRF model, the developed parallel computing solutions can be applicable to other MCRF models and other geostatistical models in sequential simulation. Python and C programming languages were used to develop the parallel computing solutions by using the multiprocessing module for parallelization with multi-core CPU(s), and CUDA and the PyCUDA module for parallelization with many-core GPUs.

The remainder of this paper is structured as follows: Section 2 gives a full description of the MCRF approach, the coMCRF model, computational analysis of the MCRF sequential simulation algorithm, and the proposed parallel computing solutions. Section 3 presents the experiments and results about the performance of the proposed parallel solutions on using the coMCRF model in land cover post-classification. Finally, this study is summarized in Section 4.

2. The coMCRF sequential simulation

2.1. The coMCRF model

The coMCRF model (Li et al. 2013, 2015) is an extension of the MCRF model (Li 2007a). An MCRF is defined as a spatial Markov chain locally-conditioned through sequential Bayesian updating on nearest data. Based on the conditional independence assumption of nearest data within a neighborhood, the MCRF local conditional probability distribution function for an unobserved location $\boldsymbol{u}_0$ is described as:

$$P[i_0(\boldsymbol{u}_0)|i_1(\boldsymbol{u}_1),\ \ldots,\ i_m(\boldsymbol{u}_m)] = \frac{P_{i_1 i_0}(\mathbf{h}_{10})\prod_{g=2}^{m} P_{i_0 i_g}(\mathbf{h}_{0g})}{\sum_{f_0=1}^{n}\left[P_{i_1 f_0}(\mathbf{h}_{10})\prod_{g=2}^{m} P_{f_0 i_g}(\mathbf{h}_{0g})\right]} \tag{1}$$

where i_0 represents the categorical class of the unobserved node at location $\boldsymbol{u}_0$; i_1 to i_m refers to the classes of the m nearest data neighbors (i.e. label-informed nodes) around $\boldsymbol{u}_0$ at corresponding locations $\boldsymbol{u}_1$ to $\boldsymbol{u}_m$ within a neighborhood; n represents the number of classes; f_0 represents all of the possible classes of the unobserved node at location $\boldsymbol{u}_0$; and finally $P_{i_0 i_g}(\mathbf{h}_{0g})$ represents a specific transition probability over the separation distance $\mathbf{h}_{0g}$ from class i_0 at location $\boldsymbol{u}_0$ to class i_g at location $\boldsymbol{u}_g$, which can be drawn from a transiogram model $P_{i_0 i_g}(\mathbf{h})$ (Li 2007b).

The transiogram was proposed as a unique name for transition probability-lag curves and functions and characterized as a generalized two-point correlation measure for categorical data by Li (2007b). A transiogram is theoretically a transition probability function of a separation distance:

$$P_{ij}(\mathbf{h}) = P[\mathrm{Z}(\boldsymbol{u}+\mathbf{h}) = j|\mathrm{Z}(\boldsymbol{u}) = i] \tag{2}$$

where $P_{ij}(\mathbf{h})$ represents a transition probability over the separation distance $\mathbf{h}$; i and j refer to the specific classes of random variables $\mathrm{Z}(\boldsymbol{u})$ and $\mathrm{Z}(\boldsymbol{u}+\mathbf{h})$ at locations $\mathbf{u}$ and $\boldsymbol{u}+\mathbf{h}$, respectively; and $\mathbf{h}$ indicates the separation distance between locations $\boldsymbol{u}$ and $\boldsymbol{u}+\mathbf{h}$. Transiogram models can be directly derived from sample data through experimental transiograms. Main studies related to transiograms can be seen in Schwarzacher (1969), Luo (1993), Carle and Fogg (1996, 1997), Li (2007b), Li and Zhang (2010), and Li, Zhang, and Dey (2012).

To approximately satisfy the requirement of the conditional independence of nearest data, the MCRF sequential simulation algorithm for implementing the MCRF model uses a quadrant (one nearest datum per quadrant) neighborhood (Li and Zhang 2007), which is similar to the quadrant search neighborhood strategy used for declustering (Klinger 1971; Isaaks and Sarivastava 1989).

The coMCRF model estimates the local conditional probability distribution by considering the co-located datum of a covariate field (i.e. a pre-classified image) when it is applied for improving land cover classification. Therefore, the coMCRF model for land cover post-classification is described as:

$$P[i_0(\boldsymbol{u}_0)|i_1(\boldsymbol{u}_1),\ \ldots,\ i_4(\boldsymbol{u}_4);\ r_0(\boldsymbol{u}_0)] = \frac{Q_{i_0 r_0}P_{i_1 i_0}(\mathbf{h}_{10})\prod_{g=2}^{4} P_{i_0 i_g}(\mathbf{h}_{0g})}{\sum_{f_0=1}^{n}\left[Q_{f_0 r_0}P_{i_1 f_0}(\mathbf{h}_{10})\prod_{g=2}^{4} P_{f_0 i_g}(\mathbf{h}_{0g})\right]} \tag{3}$$

where $Q_{i_0 r_0}$ refers to the cross-field transition probability from class i_0 at location $\boldsymbol{u}_0$ in the primary field being simulated to class r_0 at the co-location in the auxiliary field (here the pre-classified image used in land cover post-classification) (Li et al. 2015).

2.2. Execution of the coMCRF model for land cover post-classification

To execute the coMCRF model for land cover post-classification, transiogram models and the cross-field transition probability matrix are required (Li et al. 2015). Transiogram models represent the spatial auto and cross correlations of classes in the primary field to be simulated (Li 2007b; Li and Zhang 2010). The cross-field transition probability matrix represents the correlations between categorical classes within the primary field and the classes within the pre-classified image (Li et al. 2013).

The implementation of the coMCRF model consists of two stages, as shown in Figure 1: (1) data preparation and parameter setup, and (2) sequential simulation. The sequential class simulation process includes the following steps (Li and Zhang 2007): (1) set the random path for starting a realization; (2) visit the first node along the random path and acquire its position (i.e. coordinates); (3) search for the nearest (sample or already simulated) datum in each of the four quadrants within the pre-set search radius; (4) select a uniformly-distributed random number in the range of [0.0, 1.0); (5) fetch a value for the node from the calculated cumulative local conditional probability distribution based on Equation (3); (6) visit the next successive node along the random path and repeat the steps from (3) to (5) until all unobserved nodes are visited and a realization is generated; (7) start the next realization. In this sequential simulation process, the nearest neighbor searching for each neighborhood always considers the previously simulated nodes.

2.3. Computational analysis

Profiling enables us to find out the speed bottlenecks of a program (Gorelick and Ozsvald 2014). A computational analysis was made to profile the sequential simulation algorithm in order to identify the most time-consuming steps in executing the coMCRF model, so that those steps can be decomposed for domain parallelism or functional parallelism. A sequential simulation based on the coMCRF model was performed and profiled on a workstation (OS: Windows; CPU: Intel Xeon E3-1225 v3; Ram: 16GB), with a case study of 1000 × 1000 pixels for land cover post-classification. Similar to the findings in other studies (Vargas, Caetano, and Mata-Lima 2008; Pesquer, Cortés, and Pons 2011), our analysis finds that the quadrant nearest neighbor searching (NNS) and the sequential simulation steps consume most of the execution time (up to 95%) for generating a realization. The quadrant nearest neighbor searching and the random simulation cost approximate 43% and 52% of the total

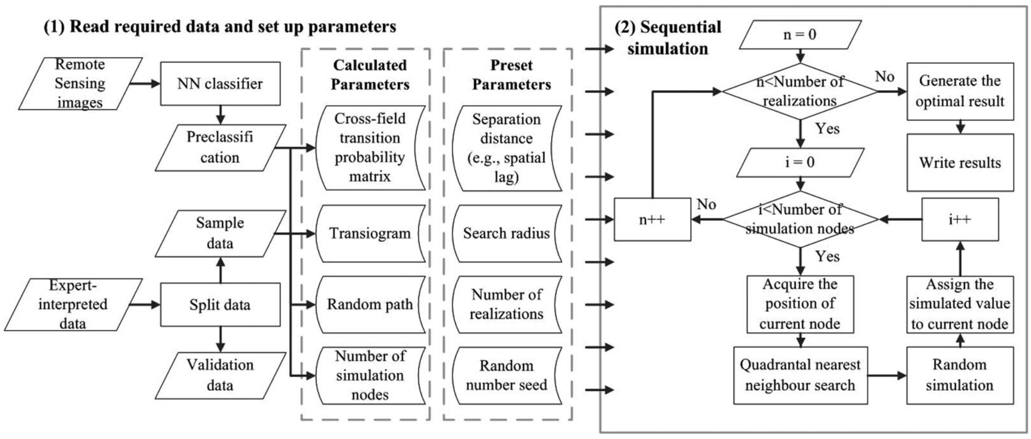
(1) Read required data and set up parameters
Remote Sensing images
NN classifier
Preclassification
Sample data
Expert-interpreted data
Split data
Validation data
Calculated Parameters
Cross-field transition probability matrix
Transiogram
Random path
Number of simulation nodes
Preset Parameters
Separation distance (e.g., spatial lag)
Search radius
Number of realizations
Random number seed
(2) Sequential simulation
n = 0
n<Number of realizations
No
Yes
Generate the optimal result
Write results
i = 0
n++
No
i<Number of simulation nodes
Yes
i++
Acquire the position of current node
Assign the simulated value to current node
Quadrantal nearest neighbour search
Random simulation

Figure 1. Flowchart of the execution of the coMCRF-based sequential simulation algorithm for land cover post-classification.

execution time, respectively. Compared to Vargas, Caetano, and Mata-Lima (2008) with respect to the time cost of the neighborhood search step, the quadrant nearest neighbor searching consumes a higher proportion of the execution time, because one quadrant neighborhood search needs to launch four independent spiral searches sequentially to identify the nearest neighbors within all quadrants. Overall, it is those two steps that limit the computational performance of the coMCRF model.

3. Parallel computing solutions suggested for the MCRF approach

In geostatistics, sequential simulation means that during the simulation process of a realization, previously simulated values for unobserved nodes along a random path are included in the sample data set for conditioning the subsequent simulation of other unobserved nodes (Goovaerts 1997; Arpat and Caers 2007; Ruggeri et al. 2013). This nature of geostatistical sequential simulation makes parallelization difficult to do. As most computers have multi-core CPUs and many-core GPUs, it is essential to properly make use of the hardware to solve computing problems (Zaccone 2015).

Mariethoz (2010) summarized three levels in parallelizing sequential simulation (i.e. realization level, path level, and node level). In this study, four parallel computing solutions were proposed to maximize the parallelization of the sequential simulation of the coMCRF model based on three parallelization functions (i.e. multicore processing, GPU-accelerated nearest neighbor searching, and GPU-accelerated approximation): (1) multicore processor parallel computing solution (MP), (2) parallel computing by GPU-accelerated nearest neighbor searching (GNNS), (3) MP with GPU-accelerated nearest neighbor searching (MP-GNNS), and (4) parallel computing by GPU-accelerated approximation (GA) and GPU-accelerated nearest neighbor searching (GA-GNNS) (see Table 1). These different solutions may be useful in different situations based on available computing facilities and actual needs.

3.1. Multicore processing

To highly parallelize a sequential simulation, the key is to minimize communication and maximize synchronization (Mariethoz 2010; Ferreirinha et al. 2015). The multicore CPU method has shown the potential for parallelizing serial programs without remodeling them (Cheng, Li, and Wang 2010). The first developed parallel computing solution for the MCRF approach, which is also the first parallelization function, is an instinctive solution at realization level – multicore simulation. In this solution, each processor core executes one realization without communications between processor cores. In general, the number of available CPU cores determines the speedup when simulating many realizations.

3.2. GPU-accelerated nearest neighbor searching

The quadrant NNS used in the MCRF approach consumes approximately 43% of the execution time of the coMCRF-based sequential class simulation. That is the second most time-consuming

Table 1. Description of four parallel computing solutions for sequential simulation using the coMCRF model.

Parallel computing solution	Number of used CPU cores	Number of used GPU devices	Parallelization function
MP	n	0	Multicore processing
GNNS	1	1	GPU-accelerated NNS
MP-GNNS	n	1	Multicore processing + GPU-accelerated NNS
GA-GNNS	1	1	GPU-accelerated approximation algorithm + GPU-accelerated NNS

'n' – the number of available CPU cores.

computation in the whole simulation process. A variety of search strategies have been used in geostatistical modeling, such as the exhaustive search, super block search, spiral search, and two-part search (Deutsch and Journel 1992). The spiral search is widely used among these strategies (Manchuk 2004). A GPU is considered an affordable compromise of computing devices with multi-threaded, multi-core processors for high computing demand (De Ravé et al. 2014). It is not easy to design a parallel algorithm for the NNS in a sequential simulation, because potential nearest neighbors are not fixed once previously simulated nodes are included in the sample data set for later conditioning; that is, the number of informed data as potential nearest neighbors for subsequent simulation of left-unobserved nodes is always changing.

The MCRF approach first used exhaustive search and then used spiral search in its quadrant neighborhood search algorithm. One quadrant NNS requires launching four independent searches sequentially to identify the nearest neighbor within each quadrant in order to approximately meet the conditional independence condition of nearest data within a neighborhood (see Equation (3)). To avoid repeating distance calculation and sorting processes, a search strategy called 'fixed-path search' was used. The parallel fixed-path search consists of three steps: (1) first defining a fixed search path from the current node to the potential nearest data node for each quadrant within a pre-set search radius (Figure 2(a)); (2) searching for the nearest data nodes along the defined paths quadrant by quadrant on GPU devices, until a data node (i.e. a sampled or previously simulated node) is found or the search reaches the limit of the search radius in each quadrant (Figures 2(b and c)); (3) gathering all the identified nearest neighbors in quadrants for the neighborhood. It is worth mentioning that if an identifiable anisotropy occurs in the whole study area (e.g. in subsurface layer cases), anisotropic quadrant search and anisotropic transiograms may be used to account for the anisotropic spatial patterns.

3.3. GPU-accelerated approximation algorithm

Approximation algorithm has been used in geostatistical sequential simulation (Dimitrakopoulos and Luo 2004; Ferreirinha et al. 2015). Rather than simulating a single node per cycle, the GA algorithm here simulates a group of nodes at a time based on the assumption that the nodes within each group partitioning a random path are relatively independent of each other. Therefore, theoretically, it is the group size of nodes that determines both the degree of uncertainty and the speedup of a sequential simulation. A larger group size of nodes leads to more improvement in computing performance but less spatial uncertainty among realizations, and vice versa (Rasera, Machado, and Costa 2015).

Inspired by the relaxation strategy in Ferreirinha et al. (2015), we applied a GA algorithm to the coMCRF-based sequential simulation by partitioning the random path for a realization based on the assumption that the unobserved nodes (to be simulated) within each partitioned section (i.e. group of nodes) along the random path are approximately independent of each other (i.e. these nodes are not clustered together, and they are only dependent locally on the nearest data excluding themselves). Such an assumption is reasonable because the nodes in each partitioned section are randomly distributed in the entire study area and usually located with a large separation distance. In fact, as the number of simulated data increases, the nodes to be simulated in each partitioned section along the random path tend to be separated by more data nodes and have more similar quadrant search results between nonparallel sequential simulation and the GA simulation (see Figure 3). The GA algorithm for one realization includes six steps (see Figure 4): (1) generate a random path visiting unobserved nodes and all random numbers for simulating a realization at a time; (2) partition the unobserved nodes along the random path into a number of groups (with n nodes for each group), (3) transfer all node groups to the GPU device at once; (4) run the GNNS for all the node groups; (5) simulate one group of nodes by n threads (i.e. the number of nodes n in a group is dependent on the number of threads in each block) at a time using the pre-set random numbers and update the simulating array; (6) then go through steps (3) to (5) to simulate the next group of nodes, till all groups are

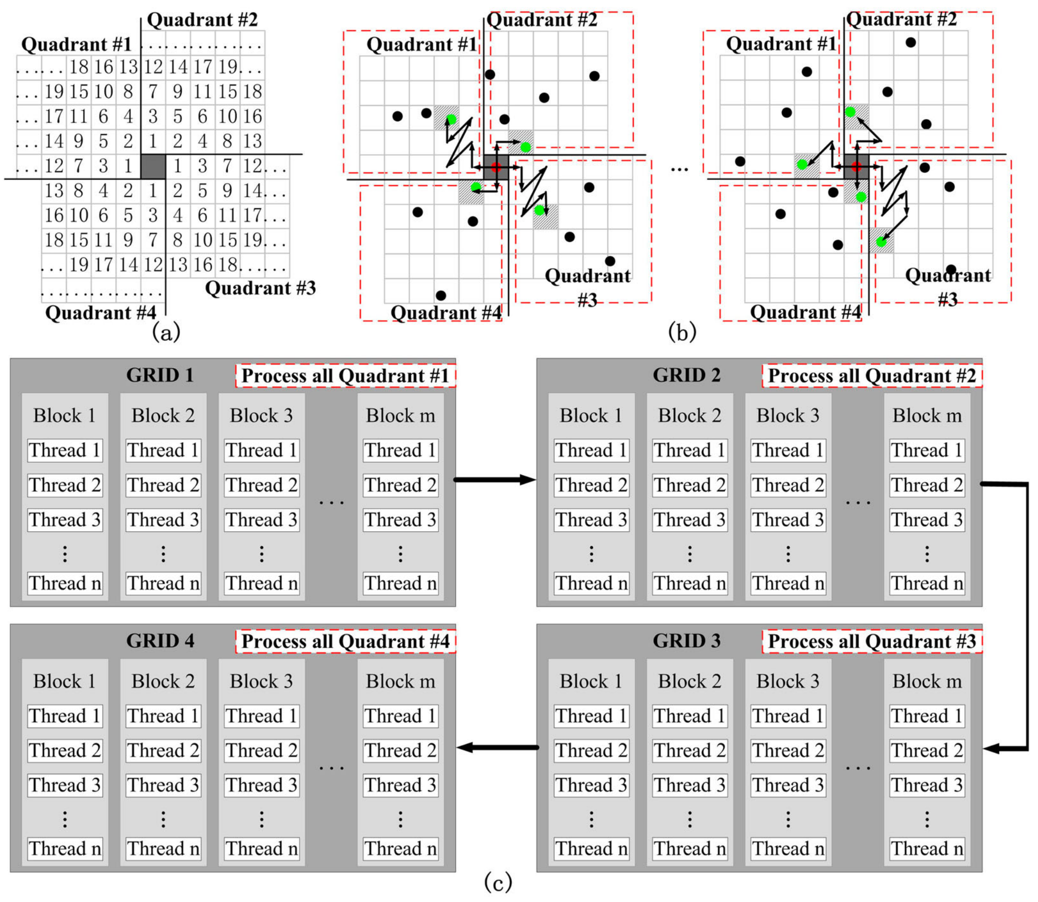

Figure 2. GPU-accelerated NNS. (a) predefined fixed search path for each quadrant, where numbers in each quadrant represent the searching sequential order; (b) quadrant nearest neighbor searching for nearest data nodes; (c) running the quadrant NNS quadrant by quadrant on GPU devices.

simulated and a realization is finished; (7) transfer simulated values back to the CPU so that they can be gathered and then written into results.

4. Experiments and results

4.1. Experiments

To verify the efficiencies of the proposed solutions, experiments in land cover post-classification based on the coMCRF model were conducted on a Haswell compute node, which is equipped with 2× Xeon E5-2690 v3 totaling 12× cores for each, 128 GB RAM, and 2× NVIDIA Tesla K40 m totaling 2880 CUDA cores for each. The coMCRF-based sequential simulation was performed with each computing solution five times in order to smooth the variance of execution time (Table 2). In land cover post-classification, a scene of 1000 × 1000 pixels with a spatial resolution of 30 m, which covers the northeastern part of Wuhan, China, with an area of 90,000 ha, was clipped from a Landsat 8 OLI image acquired on September 17, 2013. For expert-interpreted data, 4021 data points were interpreted by expert judgment based on professional insight, through integrating information from multiple sources. 80% of the expert-interpreted data points (data distribution among classes: built-up area – 34.65%, farmland – 31.01%, woodland – 4.42%, waterbody –

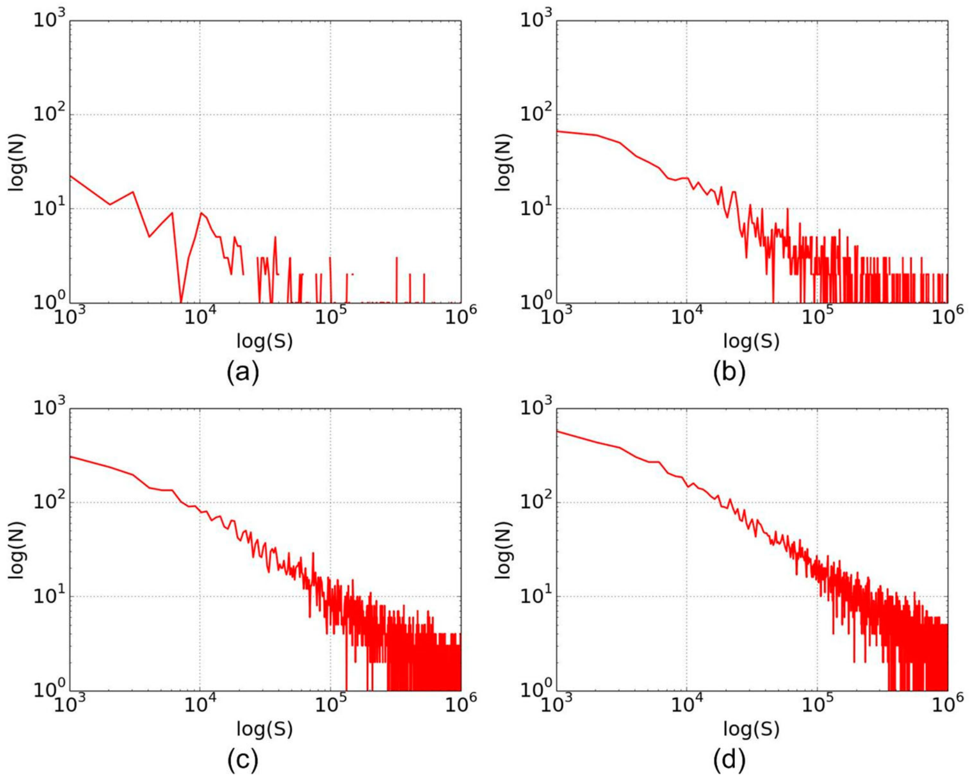

Figure 3. The sum of difference in nearest search results every 1024 nodes between nonparallel sequential simulation and GPU-accelerated approximation (GA) simulation, starting from the 1st node to the 1,000,000th node (left to right) with 32 nodes for each group (a), 128 nodes for each group (b), 512 nodes for each group (c), and 1024 nodes for each group (d). *N* denotes the number of different nearest nodes between a serial path and a partitioned path. *S* denotes the sequence number of simulation for each group. The nearest search result of GA simulation with 32 nodes for each group is much more similar to that of nonparallel sequential simulation than is that of GA simulation with 1024 nodes for each group. As the number of simulated data increases, the difference of nearest search results between nonparallel sequential simulation and GA simulation decreases.

27.65%, bare land – 2.27%) were randomly selected as conditioning sample data (i.e. hard data) in sequential simulation, and the remaining 20% (data distribution among classes: built-up area – 34.74%, farmland – 31.02%, woodland – 4.34%, waterbody – 27.54%, bare land – 2.36%) were used for validation (i.e. accuracy assessment) (Figure 5). First, the clipped image was pre-classified by a Neural Network classifier (Figure 1). Then it was post-classified by the coMCRF model, with a neighborhood search radius of 50 nodes and all computing solutions. A series of experiments were performed to test the computational efficiency of the four parallel computing solutions. Different numbers of cores ranging from 2 to 24 on a CPU were considered for the MP and MP-GNNS, and different numbers of threads for each block ranging from 32 to 1024 on a GPU device were considered for GA-GNNS (Figure 6). Finally, the performance improvement was evaluated in terms of the elapsed time for generating 100 realizations and the optimal map. The quality of each simulation was evaluated in terms of overall accuracy assessment of the optimal post-classification map generated by each computing solution (note that here overall accuracy is equal to the percentage of the correctly classified nodes among the total holdout sample data nodes for validation.)

4.2. Results

Figure 6 and Table 2 show the amounts of elapsed time spent by all parallel solutions on the land cover post-classification case by assuming the elapsed time of the nonparallel solution to be

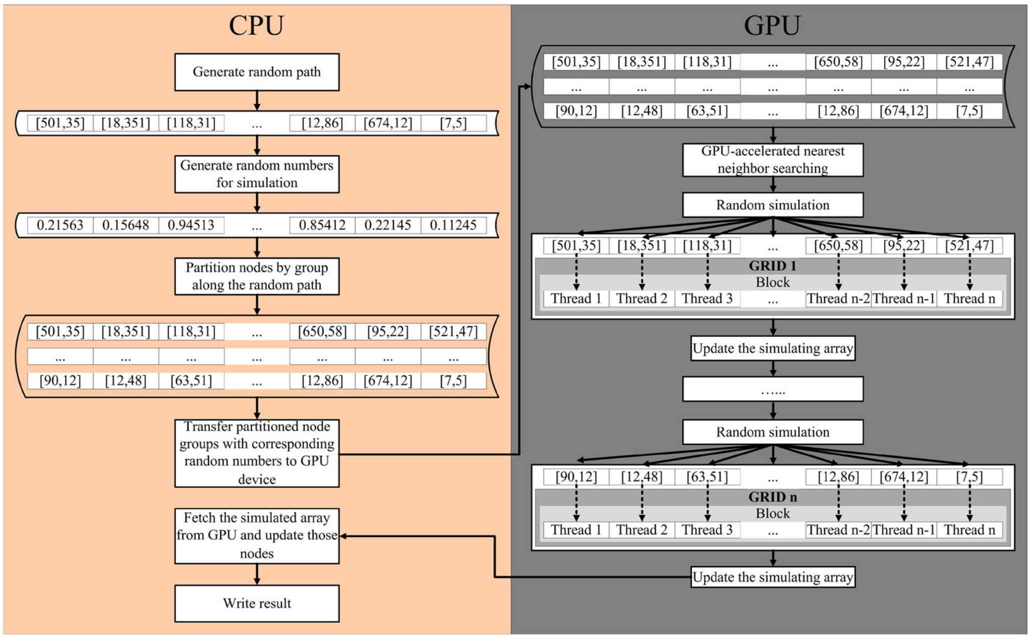
CPU
Generate random path
[501,35] [18,351] [118,31] ... [12,86] [674,12] [7,5]
Generate random numbers for simulation
0.21563 0.15648 0.94513 ... 0.85412 0.22145 0.11245
Partition nodes by group along the random path
[501,35] [18,351] [118,31] ... [650,58] [95,22] [521,47]
[90,12] [12,48] [63,51] ... [12,86] [674,12] [7,5]
Transfer partitioned node groups with corresponding random numbers to GPU device
Fetch the simulated array from GPU and update those nodes
Write result
GPU
GPU-accelerated nearest neighbor searching
Random simulation
GRID 1
Block
Thread 1 Thread 2 Thread 3 ... Thread n-2 Thread n-1 Thread n
Update the simulating array
……
Random simulation
GRID n
Block
Update the simulating array

Figure 4. Flowchart of the GPU-accelerated approximation (GA) algorithm.

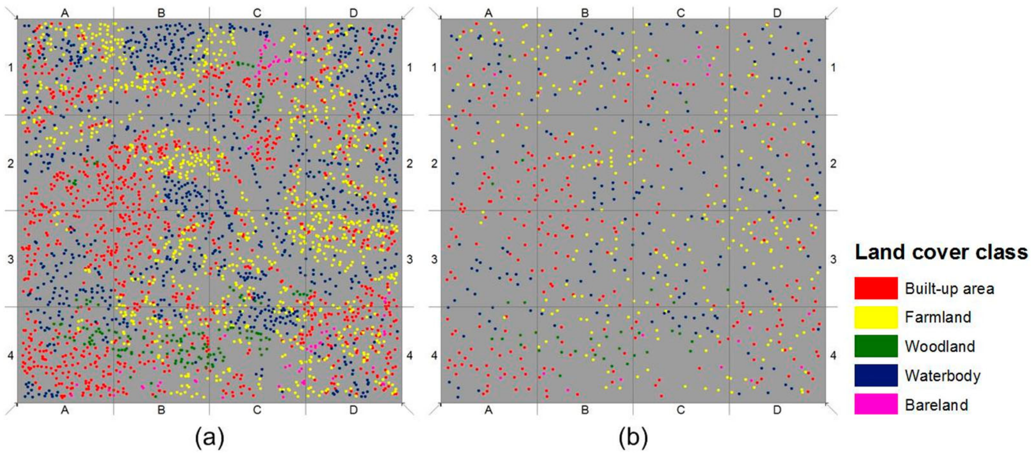

Figure 5. Expert-interpreted data: 3215 points for the coMCRF sequential simulation (a), and 806 points for validation (b).

1. The MP solution was used to test the effect of parallel computing at realization level by using multiple CPU cores. With an increasing number of CPU cores, the computational performance of the coMCRF model was improved gradually from 1.88× (at 2 CPU cores) to a peak of 16.54× (at 20 CPU cores). Because nearest neighbor searching accounts for only less than half of the elapsed time in nonparallel sequential simulation, this solution improved the total computational efficiency by 1.8×, which is significant. Further increasing the number of CPU cores (i.e. increasing the size of the parallel computing group) resulted in little benefit, which can be explained by slow spawning processes and I/O issues (Figure 6(c)). In MP-based parallel computing solutions (including MP and MP-GNNS), the parent process starts a new child process for each realization from the current parallel computing group sequentially, including both starting a fresh interpreter

Table 2. Computational efficiency of four parallel sequential simulation computing solutions compared with nonparallel sequential simulation.

Computing solution	Number of used CPU core (s)	Number of threads in each block	Test 1st elapsed time (s)	Test 2nd elapsed time (s)	Test 3rd elapsed time (s)	Test 4th elapsed time (s)	Test 5th elapsed time (s)	Average elapsed time (s)	Overall accuracy of land cover map (%)	Speedup (×)
Nonparallel sequential simulation	1	–	21,926.86	22,626.26	22,530.75	22,363.69	21,412.91	22,172.09	86.23	1.00
MP	2	–	12,111.03	13,138.85	11,769.27	10,789.75	11,182.20	11,798.22	86.23	1.88
	4	–	5975.34	6441.00	6372.41	5861.60	5701.28	6070.33	86.23	3.65
	6	–	3965.92	3970.44	3991.83	4177.79	4034.85	4028.17	86.23	5.50
	8	–	3402.45	3189.38	3348.64	3412.45	3468.26	3364.24	86.23	6.59
	10	–	2397.01	2703.85	2424.39	2406.47	2361.94	2458.73	86.23	9.02
	12	–	2410.55	2177.29	2149.17	2538.66	2210.53	2297.24	86.23	9.65
	14	–	2177.94	2060.59	2088.21	1957.11	2124.71	2081.71	86.23	10.65
	16	–	1745.93	1809.56	1721.95	1862.36	1776.68	1783.30	86.23	12.43
	18	–	1569.24	1566.69	1571.58	1548.76	1627.62	1576.78	86.23	14.06
	20	–	1299.77	1324.20	1385.40	1389.83	1302.77	1340.39	86.23	16.54
	22	–	1461.46	1308.17	1291.25	1348.58	1335.48	1348.99	86.23	16.44
	24	–	1305.35	1372.29	1303.26	1336.48	1344.00	1332.28	86.23	16.64
GNNS	1	512	11,315.05	12,998.80	13,144.30	11,310.49	12,918.27	12,337.38	86.23	1.80
MP-GNNS	2	512	5639.09	6030.38	5686.50	5499.23	6057.15	5782.47	86.23	3.83
	4	512	2990.53	2991.42	3202.00	2949.77	3317.47	3090.24	86.23	7.17
	6	512	2211.91	2285.30	2361.24	2145.18	2337.15	2268.16	86.23	9.78
	8	512	1731.10	1749.73	1713.63	1905.31	1742.60	1768.47	86.23	12.54
	10	512	1480.80	1442.48	1443.86	1462.00	1408.57	1447.54	86.23	15.32
	12	512	1308.29	1305.70	1343.95	1387.34	1357.18	1340.49	86.23	16.54
	14	512	1257.02	1249.24	1263.10	1293.45	1241.84	1260.93	86.23	17.58
	16	512	1158.48	1138.88	1196.53	1169.09	1158.72	1164.34	86.23	19.04
	18	512	1099.00	1065.50	1046.95	1099.59	1082.66	1078.74	86.23	20.55
	20	512	949.13	1035.91	978.11	958.50	948.14	973.96	86.23	22.76
	22	512	1000.33	985.17	1010.92	989.85	1047.11	1006.68	86.23	22.03
	24	512	1010.57	1002.86	1022.83	1076.91	1010.35	1024.70	86.23	21.64
GA-GNNS	1	32	479.01	497.68	508.60	480.21	499.82	493.06	85.98	44.97
	1	64	357.4	375.02	386.47	365.37	362.20	369.29	85.98	60.04
	1	128	301.78	301.26	310.19	299.92	320.57	306.74	86.10	72.28
	1	256	264.83	265.66	279.79	277.84	287.40	275.10	85.98	80.60
	1	512	251.56	282.46	263.23	254.59	271.26	264.62	86.10	83.79
	1	1024	262.09	282.40	264.88	270.54	282.38	272.46	85.86	81.38

'–': indicated not applicable.

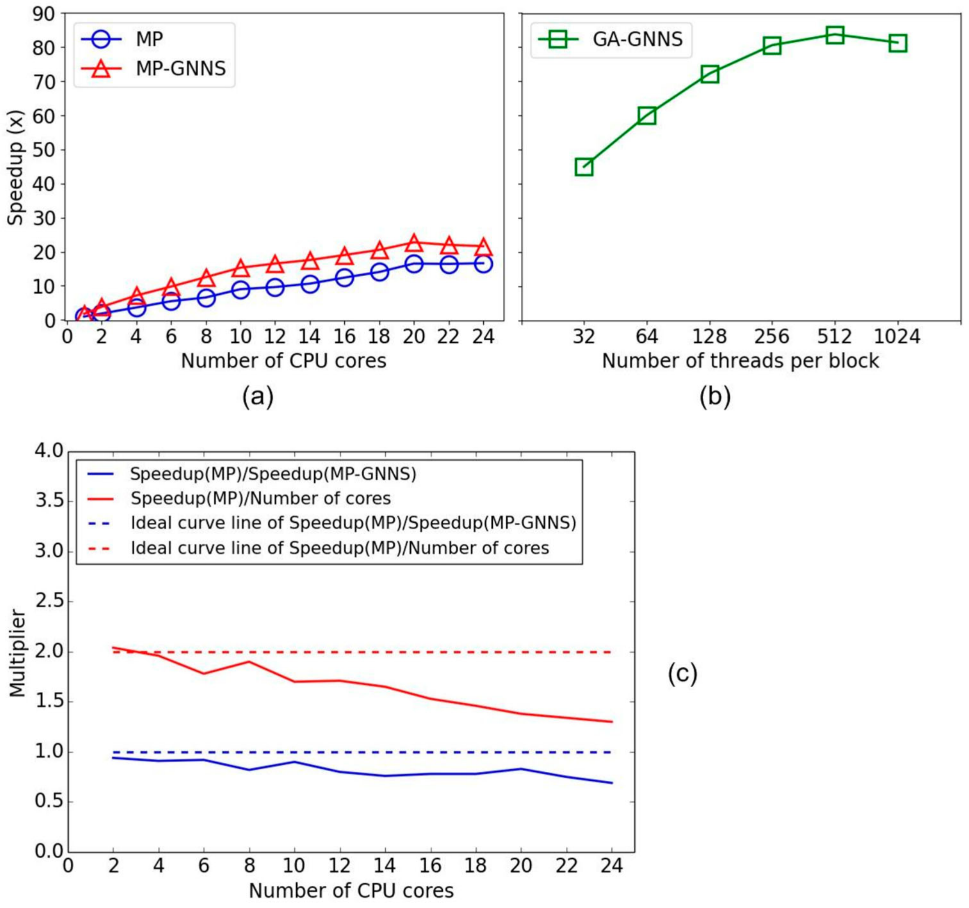

Figure 6. Speedup comparison: (a) speedup change of computational efficiency with an increasing number of CPU cores for multi-core solutions (MP and MP-GNNS); (b) speedup change of computational efficiency with an increasing number of threads for the solution with GA algorithm (GA-GNNS); (c) change in the ratio of MP-GNNS speedup to MP speedup and change in the ratio of MP speedup to the number of CPU cores.

process and inheriting necessary resources from the parent process. The program cannot start simulating the next group until the last realization in the current group is completed. As a result, increasing the number of CPU cores to some extent may make the total execution time of a stochastic simulation ascend. This impact becomes worse for MP-GNNS solutions because data transferring between a CPU host and a GPU device for each realization also costs time (Figure 6(c)). The GNNS solution was used to examine the effect of the parallel nearest neighbor searching at node level. More practical parallel computing solutions are MP-GNNS and GA-GNNS. The MP-GNNS solution combined the MP solution and the GNNS solution, and it obtained an optimal speedup at 22.76× when 20 CPU cores were used. The GA-GNNS solution combined the GA algorithm and the GNNS solution, both realizing parallel computing at node level. With GA, this solution obtained an optimal speedup at 83.79× when the number of threads was set to 512. These improvement rates with different parallel computing solutions may not appear ideal enough based on the number of used CPU cores or the number of used GPU threads. This is because there are some necessary nonparallel components and overhead communications in the coMCRF-based sequential simulation process. However, such improvements in computation speed, especially the speedup made by the GA-GNNS solution, are already very helpful to the application of the coMCRF model to land cover post-classification.

The GA-GNNS solution is faster (83× speedup) than the MP-GNNS solution. This should be attributed to the effect of the GA algorithm. Although there is no parallel computing at realization level, the use of GA at node level within a realization largely increased the computational speed for producing a realization and thus reduced the total computational time used for performing a land cover post-classification (generating 100 realizations and the optimal map). In parallel computing, managing the parallelism and load balancing are the two keys to ensure high performance. For example, the speedup increases as the group size increases in the GA algorithm (Figure 6(b)). However, the speedup decreases after the group size becomes too large (here ≥1024). This is because of the load balancing problem that a very large group size would cause an unevenly distributed workload in a GPU device. Figure 6(b) indicates that load balancing tends to be optimal when 512 nodes per group are used in the approximation algorithm in our simulation case.

In the experiment, GNNS components cost around 2.4% of elapsed time with 280 MB data being transferred each time, and GA cost around 5.9% of elapsed time with 1396 MB data being transferred. These facts indicate that the GA-GNNS solution did not suffer serious GPU memory and GPU device data transfer issues in this study, which are common bottlenecks in GPU-accelerated algorithms. The main reasons for this may be: (1) a relatively small study area (1000 × 1000 pixels), (2) proper implementation of GA-GNNS in accordance with GPU computing architectures (such as reducing data transfer and increasing the use of shared memory), and (3) the use of lightweight data types. Although this study did not examine the performance of the GA-GNNS solution with a very large image, the GA-GNNS solution might be applied to process a much larger image with a proper data partitioning and data streaming scheme based on the widely used 'divide-and-conquer' strategy without much modification of the code of the GA-GNNS solution. However, processing extremely large images is often difficult and time-consuming due to the limitations of software and computer memory.

Using multiple CPU cores and GPU-accelerated NNS (i.e. the MP, GNNS, and MP-GNNS solutions) did not cause loss in overall accuracies of post-classified land cover maps (Table 2), because the data dependency relationships in sequential simulation processes remained unchanged. However, the GA algorithm used an assumption that the nodes in each partitioned section of a predefined random path are independent of each other. Such an assumption is not absolutely true, because occasionally there may be a few cases in a partitioned section that two unobserved nodes are located very closely and thus impact each other. To evaluate whether this assumption causes accuracy loss in simulated results (i.e. post-classified maps), the quality of the simulated results was assessed and visualized in Table 2 and Figure 7. One can see that the overall accuracies (85.86%–86.10%) of post-classified land cover maps using the GA algorithm decreased very little (0.13%–0.37%), with little difference depending on the number of threads (i.e. the size of partitioned node groups). Such a minor difference in overall accuracy is visually reflected on the difference maps, as shown in Figure 7(b–g). When the number of threads per block is relatively small (here ≤128), the land cover post-classification maps generated by the GA-GNNS solution are basically the same as the post-classification map produced by the nonparallel solution, except for some minor differences along the edges of the maps where the search area for nearest neighbors is incomplete and thus simulation is more sensitive to data nodes. However, internal differences between land cover post-classification maps with and without using GA increase as the number of threads per block increases (Figure 7(b–g) and Figure 8(c–h)).

5. Conclusions

The MCRF approach is a promising geostatistical approach for modeling categorical spatial variables in multiple dimensions. However, the MCRF approach tends to be computationally costly, because it highly involves sequential simulation processes as a Markov chain-based approach. To tackle this computational issue, this study proposed and examined four parallel computing

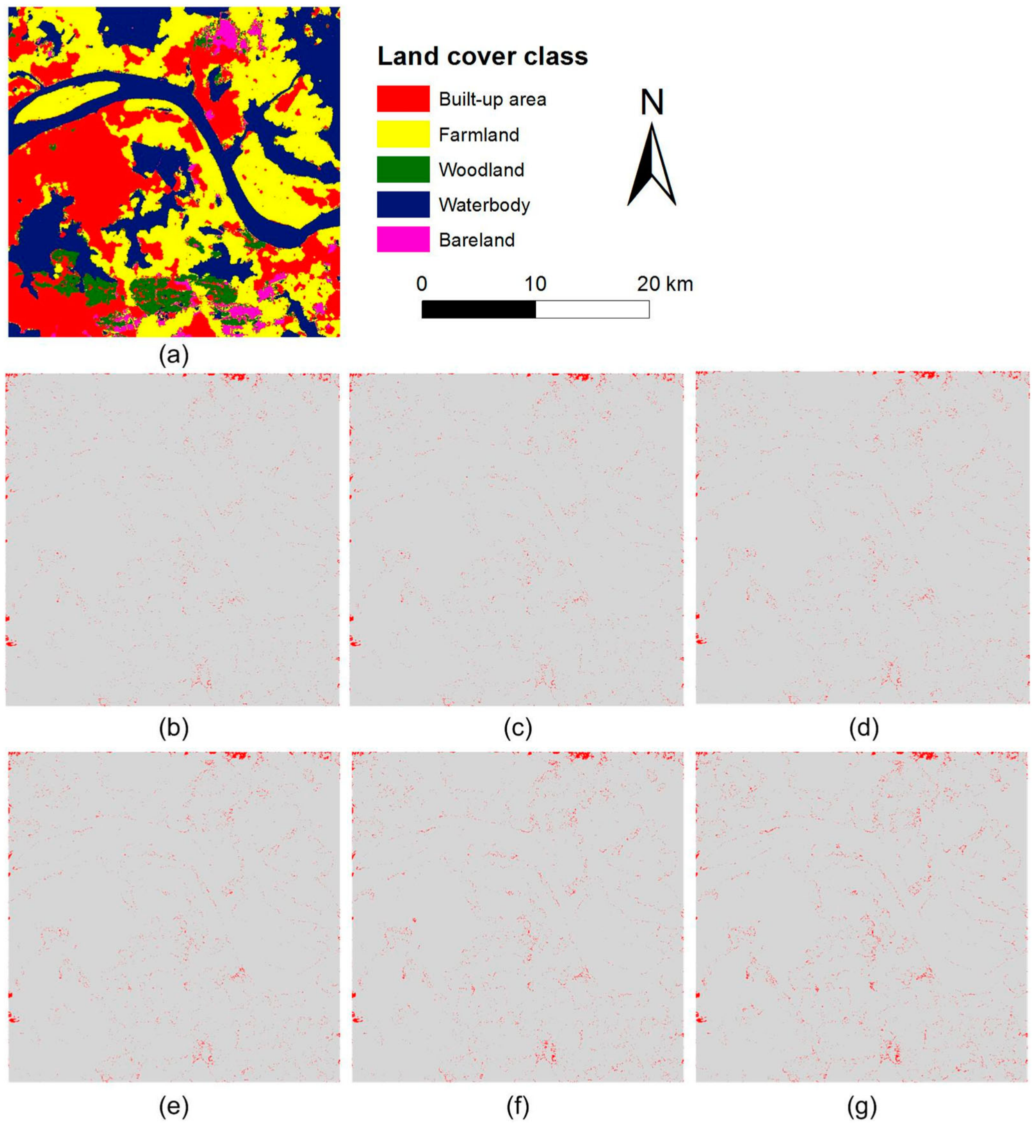

Figure 7. Comparison of land cover post-classification maps with and without using the GA algorithm. Land cover post-classification map by nonparallel sequential simulation (a). Difference maps (light gray color: the same; red color: different) between nonparallel sequential simulation (a) and parallel sequential simulation (GA-GNNS) with 32 threads (b), 64 threads (c), 128 threads (d), 256 threads (e), 512 threads (f), and 1024 threads per block (g).

solutions for MCRF sequential simulation to improve the computing efficiency of the MCRF approach. Testing of the proposed parallel computing solutions was performed on a specific application of the MCRF approach – land cover post-classification, which uses the coMCRF model to conduct cosimulation on expert-interpreted sample data and pre-classified land cover data to improve classification accuracy. The proposed parallel computing solutions for MCRF sequential simulation are MP, GNNS, MP-GNNS, and GA-GNNS. Our testing showed that significant improvement in computational efficiency can be achieved by parallel computing solutions by leveraging the power of multiple CPU cores, GPU devices, and an approximation algorithm. It was also observed that three parallel computing solutions generated the same result

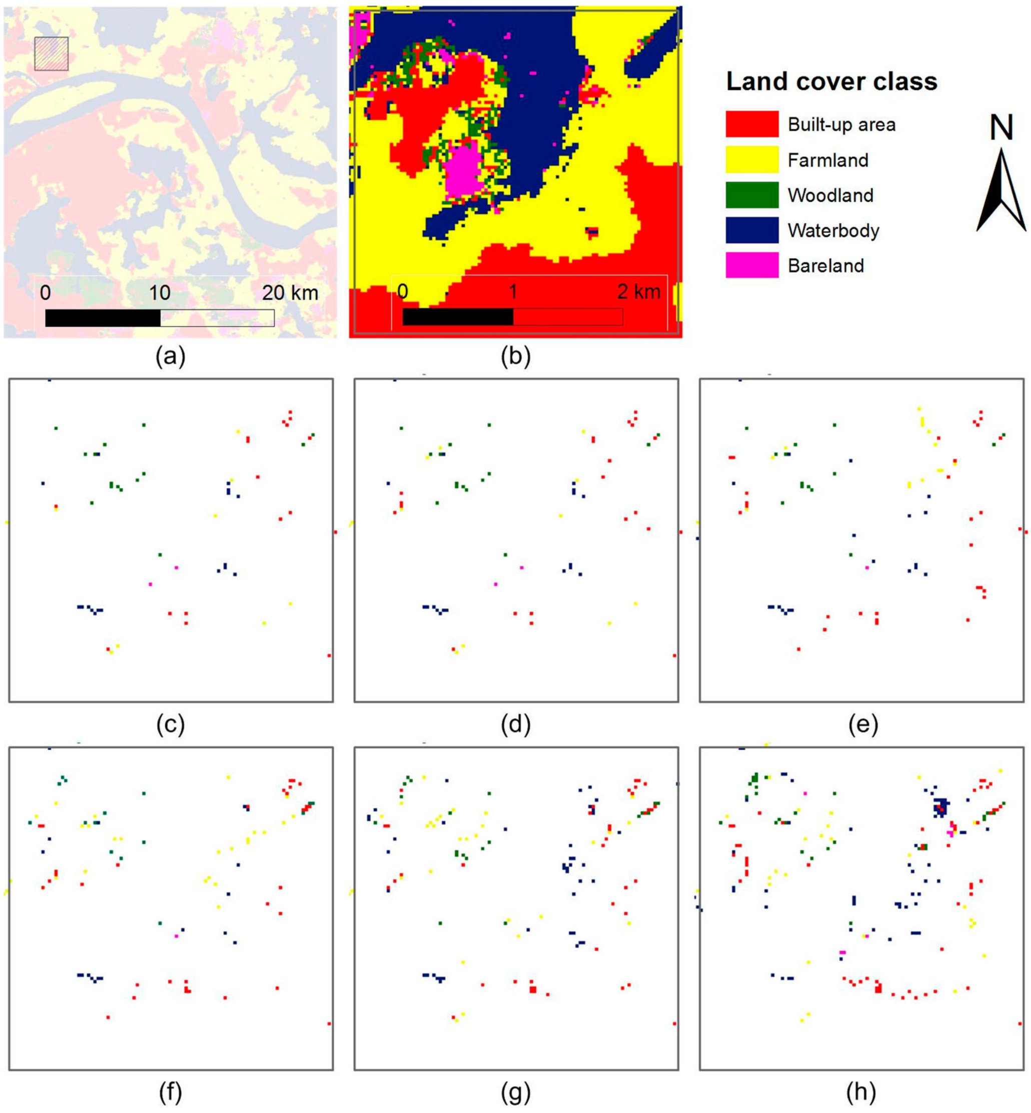

Figure 8. Enlarged comparison of land cover post-classification maps with and without using the GA algorithm. (a) The location of the enlarged area in the post-classification map. (b) Land cover post-classification map by nonparallel sequential simulation. Difference maps (white color: the same; other colors: classes of those pixels assigned by parallel sequential simulation but different from those by the nonparallel sequential simulation) between nonparallel sequential simulation and parallel sequential simulation (GA-GNNS) with 32 threads (c), 64 threads (d), 128 threads (e), 256 threads (f), 512 threads (g), and 1024 threads per block (h).

as the nonparallel solution did, but cost much less execution time. With the GA algorithm, the GA-GNNS solution is able to greatly improve the computational efficiency of the coMCRF model by 83× at a suitable number of block threads while causing little loss in overall accuracy that may be negligible based on the comparison of post-classification maps. Particularly, compared to multi-core solutions, which usually require more than 2 cores in a CPU, the proposed parallel computing solutions that only rely on a GPU are more practical and portable. Based on available computing facilities and actual needs, users may choose one of them in real applications.

Acknowledgements

The authors would like to thank Krista Rogers for her helpful reviews of this manuscript. The authors also would like to thank the UConn HPC team, especially Luke Malinowski, for their help in using the high performance computer at UConn. Authors have the sole responsibility of all viewpoints presented in this paper.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported in part by the U.S. National Science Foundation [grant number 1414108]; Division of Behavioral and Cognitive Sciences.

ORCID

Weixing Zhang http://orcid.org/0000-0001-8253-3932
Weidong Li http://orcid.org/0000-0002-4558-3292
Chuanrong Zhang http://orcid.org/0000-0001-5249-141X

References

Armstrong, M. P., and R. J. Marciano. 1997. "Massively Parallel Strategies for Local Spatial Interpolation." *Computers & Geosciences* 23 (8): 859–867. doi:10.1016/S0098-3004(97)00058-7.

Arpat, G. B., and J. Caers. 2007. "Conditional Simulation with Patterns." *Mathematical Geology* 39 (2): 177–203. doi:10.1007/s11004-006-9075-3.

Bierkens, M. F. P., and P. A. Burrough. 1993. "The Indicator Approach to Categorical Soil Data: I. Theory." *Journal of Soil Science* 44 (2): 361–368. doi:10.1111/j.1365-2389.1993.tb00458.x.

Carle, S. F., and G. E. Fogg. 1996. "Transition Probability-Based Indicator Geostatistics." *Mathematical Geology* 28 (4): 453–476.

Carle, S. F., and G. E. Fogg. 1997. "Modeling Spatial Variability with one- and Multi-Dimensional Continuous Markov Chains." *Mathematical Geology* 29: 891–918.

Chao, F., Y. Chongjun, C. Zhuo, Y. Xiaojing, and G. Hantao. 2011. "Parallel Algorithm for Viewshed Analysis on a Modern GPU." *International Journal of Digital Earth* 4 (6): 471–486. doi:10.1080/17538947.2011.555565.

Cheng, T. 2013. "Accelerating Universal Kriging Interpolation Algorithm Using CUDA-Enabled GPU." *Computers & Geosciences* 54: 178–183. doi:10.1016/j.cageo.2012.11.013.

Cheng, T., D. Li, and Q. Wang. 2010. "On Parallelizing Universal Kriging Interpolation Based on OpenMP." Ninth international symposium on distributed computing and applications to business engineering and science (DCABES), 36–39. doi:10.1109/DCABES.2010.14.

De Ravé, E. G., F. J. Jiménez-Hornero, A. B. Ariza-Villaverde, and J. M. Gómez-López. 2014. "Using General-Purpose Computing on Graphics Processing Units (GPGPU) to Accelerate the Ordinary Kriging Algorithm." *Computers & Geosciences* 64: 1–6. doi:10.1016/j.cageo.2013.11.004.

Deutsch, C. V., and A. G. Journel. 1992. *Geostatistical Software Library and User's Guide*. New York: Oxford University Press.

Dimitrakopoulos, R., and X. Luo. 2004. "Generalized Sequential Gaussian Simulation on Group Size ν and Screen-Effect Approximations for Large Field Simulations." *Mathematical Geology* 36 (5): 567–591. doi:10.1023/B:MATG.0000037737.11615.df.

Du, Z., Y. Gu, C. Zhang, F. Zhang, R. Liu, J. Sequeira, and W. Li. 2017. "ParSymG: A Parallel Clustering Approach for Unsupervised Classification of Remotely Sensed Imagery." *International Journal of Digital Earth* 10 (5): 471–489. doi:10.1080/17538947.2016.1229818.

Ferreirinha, T., R. Nunes, L. Azevedo, A. Soares, F. Pratas, P. Tomás, and N. Roma. 2015. "Acceleration of Stochastic Seismic Inversion in OpenCL-Based Heterogeneous Platforms." *Computers & Geosciences* 78: 26–36. doi:10.1016/j.cageo.2015.02.005.

Goovaerts, P. 1997. *Geostatistics for Natural Resources Evaluation*. New York: Oxford University Press.

Goovaerts, P. 2002. "Geostatistical Modelling of Spatial Uncertainty Using p-field Simulation with Conditional Probability Fields." *International Journal of Geographical Information Science* 16 (2): 167–178. doi:10.1080/13658810110099125.

Gorelick, M., and I. Ozsvald. 2014. *High Performance Python*. Sebastopol: O'Reilly Media, Inc.

Ingram, B., and D. Cornford. 2010. "Parallel Geostatistics for Sparse and Dense Datasets." *geoENV VII-Geostatistics for Environmental Applications* 16: 371–381. doi:10.1007/978-90-481-2322-3_32.

Isaaks, E. H., and R. Sarivastava. 1989. *An Introduction to Applied Geostatistics*. New York: Oxford University Press.

Juang, K. W., Y. S. Chen, and D. Y. Lee. 2004. "Using Sequential Indicator Simulation to Assess the Uncertainty of Delineating Heavy-Metal Contaminated Soils." *Environmental Pollution* 127: 229–238. doi:10.1016/j.envpol.2003.07.001.

Klinger, A. 1971. "Patterns and Search Statistics." *Optimizing Methods in Statistics*, 303–337. doi:10.1016/B978-0-12-604550-5.50019-5.

Lark, R. M. 2000. "Regression Analysis with Spatially Autocorrelated Error: Simulation Studies and Application to Mapping of Soil Organic Matter." *International Journal of Geographical Information Science* 14 (3): 247–264. doi:10.1080/136588100240831.

Lee, Y. M., and J. H. Ellis. 2000. "Turning Bands Method and its Application to Geostatistical Analysis of Groundwater Contaminant Concentration Fields." *Journal of the Chinese Institute of Environmental Engineering* 10 (1): 1–12.

Li, W. 2007a. "Markov Chain Random Fields for Estimation of Categorical Variables." *Mathematical Geology* 39 (3): 321–335. doi:10.1007/s11004-007-9081-0.

Li, W. 2007b. "Transiograms for Characterizing Spatial Variability of Soil Classes." *Soil Science Society of America Journal* 71 (3): 881–893. doi:10.2136/sssaj2005.0132.

Li, Z., M. E. Hodgson, and W. Li. 2016. "A General-Purpose Framework for Parallel Processing of Large-Scale LiDAR Data." *International Journal of Digital Earth*, 1–22. doi:10.1080/17538947.2016.1269842.

Li, X., T. Huang, D. T. Lu, and C. Niu. 2014. "Accelerating Experimental High-Order Spatial Statistics Calculations Using GPUs." *Computers & Geosciences* 70: 128–137. doi:10.1016/j.cageo.2014.05.012.

Li, W., and C. Zhang. 2007. "A Random-Path Markov Chain Algorithm for Simulating Categorical Soil Variables From Random Point Samples." *Soil Science Society of America Journal* 71 (3): 656–668. doi:10.2136/sssaj2006.0173.

Li, W., and C. Zhang. 2010. "Linear Interpolation and Joint Model Fitting of Experimental Transiograms for Markov Chain Simulation of Categorical Spatial Variables." *International Journal of Geographical Information Science* 24 (6): 821–839. doi:10.1080/13658810903127991.

Li, W., C. Zhang, and D. K. Dey. 2012. "Modeling Experimental Cross-Transiograms of Neighboring Landscape Categories with the Gamma Distribution." *International Journal of Geographical Information Science* 26 (4): 599–620. doi:10.1080/13658816.2011.603336.

Li, W., C. Zhang, D. K. Dey, and M. R. Willig. 2013. "Updating Categorical Soil Maps Using Limited Survey Data by Bayesian Markov Chain Cosimulation." *The Scientific World Journal*. doi:10.1155/2013/587284.

Li, W., C. Zhang, M. R. Willig, D. K. Dey, G. Wang, and L. You. 2015. "Bayesian Markov Chain Random Field Cosimulation for Improving Land Cover Classification Accuracy." *Mathematical Geosciences* 47 (2): 123–148. doi:10.1007/s11004-014-9553-y.

Liu, J., D. Feld, Y. Xue, J. Garcke, T. Soddemann, and P. Pan. 2016. "An Efficient Geosciences Workflow on Multi-Core Processors and GPUs: a Case Study for Aerosol Optical Depth Retrieval From MODIS Satellite Data." *International Journal of Digital Earth* 9 (8): 748–765. doi:10.1080/17538947.2015.1130087.

Luo, J. 1993. "Transition Probability Approach to Statistical Analysis of Spatial Qualitative Variables in Geology." In *Geologic Modeling and Mapping*, edited by Andrea Förster and Daniel F. Merriam, 281–299. New York: Plenum Press.

Manchuk, J. 2004. "Search Strategies for Geostatistical Simulation of Unstructured Grids." Accessed May 28, 2017. http://www.ccgalberta.com/ccgresources/report06/2004-102-search_unstructured_grids.pdf.

Mariethoz, G. 2010. "A General Parallelization Strategy for Random Path Based Geostatistical Simulation Methods." *Computers & Geosciences* 36 (7): 953–958. doi:10.1016/j.cageo.2009.11.001.

Mei, G. 2014. "Evaluating the Power of GPU Acceleration for IDW Interpolation Algorithm." *The Scientific World Journal*. doi:10.1155/2014/171574.

Mowrer, H. T. 1997. "Propagating Uncertainty Through Spatial Estimation Processes for Old-Growth Subalpine Forests Using Sequential Gaussian Simulation in GIS." *Ecological Modelling* 98: 73–86. doi:10.1016/S0304-3800(96)01938-2.

Nunes, R., and J. A. Almeida. 2010. "Parallelization of Sequential Gaussian, Indicator and Direct Simulation Algorithms." *Computers & Geosciences* 36 (8): 1042–1052. doi:10.1016/j.cageo.2010.03.005.

Peredo, O., J. M. Ortiz, and J. R. Herrero. 2015. "Acceleration of the Geostatistical Software Library (GSLIB) by Code Optimization and Hybrid Parallel Programming." *Computers & Geosciences* 85: 210–233. doi:10.1016/j.cageo.2015.09.016.

Pesquer, L., A. Cortés, and X. Pons. 2011. "Parallel Ordinary Kriging Interpolation Incorporating Automatic Variogram Fitting." *Computers & Geosciences* 37 (4): 464–473. doi:10.1016/j.cageo.2010.10.010.

Rasera, L. G., P. L. Machado, and J. F. C. Costa. 2015. "A Conflict-Free, Path-Level Parallelization Approach for Sequential Simulation Algorithms." *Computers & Geosciences* 80: 49–61. doi:10.1016/j.cageo.2015.03.016.

Ruggeri, P., J. Irving, E. Gloaguen, and K. Holliger. 2013. "Regional–Scale Integration of Multiresolution Hydrological and Geophysical Data Using a Two-Step Bayesian Sequential Simulation Approach." *Geophysical Journal International* 194 (1): 289–303. doi:10.1093/gji/ggt067.

Schwarzacher, W. 1969. "The Use of Markov Chains in the Study of Sedimentary Cycles." *Mathematical Geology* 1: 17–39.

Shi, X., and F. Ye. 2013. "Kriging Interpolation Over Heterogeneous Computer Architectures and Systems." *GIScience & Remote Sensing* 50 (2): 196–211. doi:10.1080/15481603.2013.793480.

Srinivasan, B. V., R. Duraiswami, and R. Murtugudde. 2010. "Efficient Kriging for Real-Time Spatio-Temporal Interpolation." Proceedings of the 20th conference on probability and statistics in the atmospheric sciences, 228–235

Srivastava, M. R. 1996. "An Overview of Stochastic Simulation." Spatial accuracy assessment in natural resources and environmental sciences: second international symposium, Fort Collins, May 21–23.

Tahmasebi, P., M. Sahimi, G. Mariethoz, and A. Hezarkhani. 2012. "Accelerating Geostatistical Simulations Using Graphics Processing Units (GPU)." *Computers & Geosciences* 46: 51–59. doi:10.1016/j.cageo.2012.03.028.

Vargas, H. S., H. Caetano, and H. Mata-Lima. 2008. "A New Parallelization Approach for Sequential Simulation." *geoENV VII-Geostatistics for Environmental Applications* 15: 489–496. doi:10.1007/978-1-4020-6448-7_40.

Wang, S., and M. P. Armstrong. 2009. "A Theoretical Approach to the Use of Cyberinfrastructure in Geographical Analysis." *International Journal of Geographical Information Science* 23 (2): 169–193. doi:10.1080/13658810801918509.

Wei, H., Y. Du, F. Liang, C. Zhou, Z. Liu, J. Yi, K. Xu, and D. Wu. 2015. "A k-d Tree-Based Algorithm to Parallelize Kriging Interpolation of Big Spatial Data." *GIScience & Remote Sensing* 52 (1): 40–57. doi:10.1080/15481603.2014.1002379.

Weissmann, G. S., and G. E. Fogg. 1999. "Multi-Scale Alluvial Fan Heterogeneity Modeled with Transition Probability Geostatistics in a Sequence Stratigraphic Framework." *Journal of Hydrology* 226: 48–65. doi:10.1016/S0022-1694(99)00160-2.

Wu, M. C., W. Y. Chang, Y. L. Chang, S. Y. Shih, C. Chu, and B. Huang. 2016. "High-Performance Adaptive Local Kriging Applied to Recovering Surface Deformation Associated with the Fault Zones." *Geoscience and Remote Sensing Symposium (IGARSS)*, 6014–6017. doi:10.1109/IGARSS.2016.7730571.

Wu, K., N. Nunan, J. W. Crawford, I. M. Young, and K. Ritz. 2004. "An Efficient Markov Chain Model for the Simulation of Heterogeneous Soil Structure." *Soil Science Society of America Journal* 68 (2): 346–351. doi:10.2136/sssaj2004.3460.

Zaccone, G. 2015. *Python Parallel Programming Cookbook*. Birmingham: Packt Publishing Ltd.

Zhang, C., and W. Li. 2008. "Regional-Scale Modeling of the Spatial Distribution of Surface and Subsurface Textural Classes in Alluvial Soils Using Markov Chain Geostatistics." *Soil Use and Management* 24 (3): 263–272. doi:10.1111/j.1475-2743.2008.00165.x.

Zhang, W., W. Li, and C. Zhang. 2016. "Land Cover Post-Classifications by Markov Chain Geostatistical Cosimulation Based on pre-Classifications by Different Conventional Classifiers." *International Journal of Remote Sensing* 37 (4): 926–949. doi:10.1080/01431161.2016.1143136.

Zhang, W., W. Li, C. Zhang, and X. Li. 2017a. "Incorporating Spectral Similarity Into Markov Chain Geostatistical Cosimulation for Reducing Smoothing Effect in Land Cover Postclassification." *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 10 (3): 1082–1095. doi:10.1109/JSTARS.2016.2596040.

Zhang, W., W. Li, C. Zhang, and W. B. Ouimet. 2017b. "Detecting Horizontal and Vertical Urban Growth From Medium Resolution Imagery and its Relationships with Major Socioeconomic Factors." *International Journal of Remote Sensing* 38 (12): 3704–3734. doi:10.1080/01431161.2017.1302113.

Zhao, Y., X. Shi, D. Yu, H. Wang, and W. Sun. 2005. "Uncertainty Assessment of Spatial Patterns of Soil Organic Carbon Density Using Sequential Indicator Simulation, a Case Study of Hebei Province, China." *Chemosphere* 59: 1527–1535. doi:10.1080/17538947.2014.904012.

Zhao, T., C. Zhang, L. Anselin, W. Li, and K. Chen. 2015. "A Parallel Approach for Improving Geo-SPARQL Query Performance." *International Journal of Digital Earth* 8 (5): 383–402. doi:10.1080/17538947.2014.904012.