DSLEUTH: A DISTRIBUTED VERSION OF SLEUTH URBAN GROWTH MODEL

Gargi Chaudhuri

Department of Geography and Earth Science
University of Wisconsin-La Crosse
1725 State St., 2022 Cowley Hall
La Crosse, WI, USA
gchaudhuri@uwlax.edu

Samantha Foley

Department of Computer Science
University of Wisconsin-La Crosse
1725 State St., 220 Wing Tech. Ctr.
La Crosse, WI, USA
sfoley@uwlax.edu

ABSTRACT

Land use change models have seen a growth in use due to the availability of multi-temporal, high resolution images, and computational power. One such model, SLEUTH, is a popular cellular automata model for studying land use change, especially urban growth. While it is widely adopted and highly regarded, the computational complexity of the brute-force parameter sweep style of execution leads to long run times, which is further exacerbated by large area and high resolution images. In this study, we developed a distributed framework for SLEUTH, called DSLEUTH, for executing and managing subsets of the parameter space on a shared memory machine. This paper describes the framework and a performance study using two different data sets. The framework yielded a 20x speed up on both datasets using 40 processes.

Keywords: SLEUTH, simulation model, distributed computing, cellular automata, urban growth model.

1 INTRODUCTION

The study of land use science has been heavily influenced by two advancements: the use of spatial simulation models, and greater availability of multi-temporal, high resolution satellite images to understand the pattern and process of land use change. Urban growth models, a type of spatial simulation models, are used to simulate and forecast urban growth. These models are used by academics and planners to: (1) evaluate the impact of policies and zoning in urban areas (Onsted and Chowdhury 2014), (2) to predict urban expansion under different growth scenarios (Barredo and Kasanko 2003; Solecki and Oliveri 2004; Chaudhuri and Clarke 2012), and (3) to evaluate the factors influencing urbanization (Jantz, Drzyzga, and Maret 2014; Tayyebi et al. 2013). Cellular automata, agent-based, or spatial econometric models (or hybrids thereof) form the basis of the majority of popular urban growth models (Agarwal et al. 2002; Pontius et al. 2008). Arguably, the cellular automata models have been the most popular because of its simplicity, its natural fit with the raster data model (Batty, Xie, and Sun 1999), and its ability to simulate the pattern and process of urban growth with a given set of transition rules (Torrens and O'Sullivan 2001).

Freely available datasets of changing urban landscapes which have medium- to high spatial resolution have heavily contributed to the understanding of how urban landscapes change over time. Using these datasets, which are rich in both space and time, is challenging. The majority of cellular automata-based geosimulation models follow the traditional sequential approach for calibration, which is computationally intensive and take long periods of time. Calibration time is directly related to the amount of data required for the specific model, the size of the input data (spatial extent and resolution), the temporal length of input data, and the computational complexity of each model (Shashidharan et al. 2016). Existing literature shows two approaches to address this problem, first by using different calibration techniques, and secondly by using

parallel and distributed computational frameworks. Use of different computational techniques, such as, artificial neural networks (Pijanowski et al. 2002), genetic algorithms (Clarke 2017), support vector machines (Rienow and Goetzke 2015), and logistic regressions (Hu and Lo 2007), helps to derive the control parameters more quickly during the calibration process. These approaches have reduced the calibration time to some extent, but these gains are offset by the computational complexity of the optimizations. However, adaptation of sequential models into parallel and distributed frameworks has been relatively more successful (Hawick, Coddington, and James 2003; Guan and Shi 2013; Clarke 2003) to decrease the computational time and scale up according to data and computational needs of the calibration problems.

The goal of the present study is to adapt a popular cellular automata-based sequential land use change model, SLEUTH, into a distributed framework. SLEUTH has been applied extensively in the geographic simulation of future planning scenarios (Clarke, Hoppen, and Gaydos 1997; Chaudhuri and Clarke 2013), and compared and examined with other models of land use change (Agarwal et al. 2002; Pontius et al. 2008). The popularity of SLEUTH is mostly attributed to the minimum data requirements, relatively simple modeling workflow, and ease of adaptation (Clarke 2008; Chaudhuri and Clarke 2013). It is a brute force parameter sweep application and uses a Monte Carlo method to simulate a random process.

In order to decrease the calibration time and increase the accuracy of the output image, we developed a distributed version of the model called DSLEUTH. Without decomposing the dataspace, DSLEUTH splits the parameter search space into concurrent runs and automates the sorting of the results from each calibration phase into the same format as SLEUTH so that users do not see a difference when using DSLEUTH. DSLEUTH greatly improves the calibration time without compromising the accuracy of the model output. This is a significant improvement over the present SLEUTH3.0 version because it allows model users to conduct regional level analysis with bigger data sizes and utilize parallel computing in geospatial problem solving. This work has been done on a shared memory, multicore computer, and we plan to extend this framework to distributed memory clusters.

2 SLEUTH MODEL BACKGROUND

SLEUTH is an open-source cellular automata based land use change simulation model that has been applied to a large number of cities around the world (Chaudhuri and Clarke 2013). Detailed list of application is available on the SLEUTH website (http://anteater.geog.ucsb.edu/gig/index.html). The Urban Growth Model (UGM) is tightly coupled with the Deltatron Land Use/Land Cover Model (DLM) to form the SLEUTH model (Clarke, Hoppen, and Gaydos 1997). SLEUTH is an acronym for the raster data layers required by the model, **S**lope, **L**and use, **E**xclusion, **U**rban extent, **T**ransportation, and **H**illshade, and simulates land use dynamics as a physical process (Clarke, Hoppen, and Gaydos 1997). The slope and the hillshaded data layers are derived from the topographic data. The hillshade layer is only used for visualization and does not play any role in determining model results. For land use, raster layers from two time periods with a consistent land use/cover classification scheme and urban layers for four time periods are required. An exclusion layer is included that restricts urban growth in locations such as wetlands, lakes and oceans. This layer can be also used for scenario development and implement zoning restrictions by using partial and/or complete exclusion (Onsted and Chowdhury 2014; Jantz, Drzyzga, and Maret 2014). The users of the model can also choose to run only the UGM sub-model, in that case only urban growth will be simulated and forecasted and land use raster layers are not required. The present study focused on the urban growth part of the model, so the following discussion will describe the UGM sub-model in detail.

The urban areas inside this model are trained by a finite set of transition rules that influence the state of changes from non-urban to urban as a set of nested loops (Silva and Clarke 2002). The model calibrates the input urban layers from four time periods in the past to derive a set of control parameters (dispersion coefficient, breed coefficient, spread coefficient, slope resistance factor, and road coefficient) that control the behavior of the system and capture the past urbanization trends of that region (Chaudhuri and Clarke 201. These parameter coefficient values are used to calculate the growth rate that determines the degree to

which each of the four growth rules (spontaneous, diffusive, organic, and road influenced growth) influences urban growth in the system (Gazulis and Clarke 2006). In addition to the initial growth rules, there is a set of secondary rules called the 'self-modification' rules, which respond to the aggregate growth rate. The growth rate is the sum of the four different types of growth (spontaneous, diffusive, organic, and road influenced growth) defined by the model for each growth cycle (Figure 1). The number of growth cycles is defined by the difference between the first and the last year of the input data. The upper and the lower threshold limits are predefined and they kick off an increase or decrease in three of the growth control parameters: dispersion, breed, and spread. If the growth rate exceeds the upper threshold, the coefficients are increased by a multiplier greater than one and the phenomenon is termed as BOOM. This increase imitates the tendency of an expanding system to grow ever more rapidly. If the growth rate falls below the lower threshold, the coefficients are decreased by a multiplier less than one and the phenomenon is termed as BUST. This causes growth to taper off as it does in a depressed or saturated system. Self-modification is important to avoid linear or exponential growth of the area in the model (Silva and Clarke 2002).

SLEUTH uses three sequential brute force calibration phases: coarse, fine, and final, to refine the set of control parameters (Silva and Clarke 2002). The Optimal SLEUTH Metric (OSM) (Dietzel and Clarke 2007) is used by the researcher at the end of each calibration phase to choose the best-fit parameter combination which are used in the next phase of calibration. The OSM function tallies each of the Monte Carlo runs stored in control_stats.log file and calculates the metric by taking a product of multiple indices generated by the model. The highest OSM values indicate the best-fit parameter combinations that captures the urban growth pattern and process most closely to trend generated from the input data (Clarke 2008). The combination of parameters with highest OSM value in the final calibration phase is used for forecasting future urban growth.

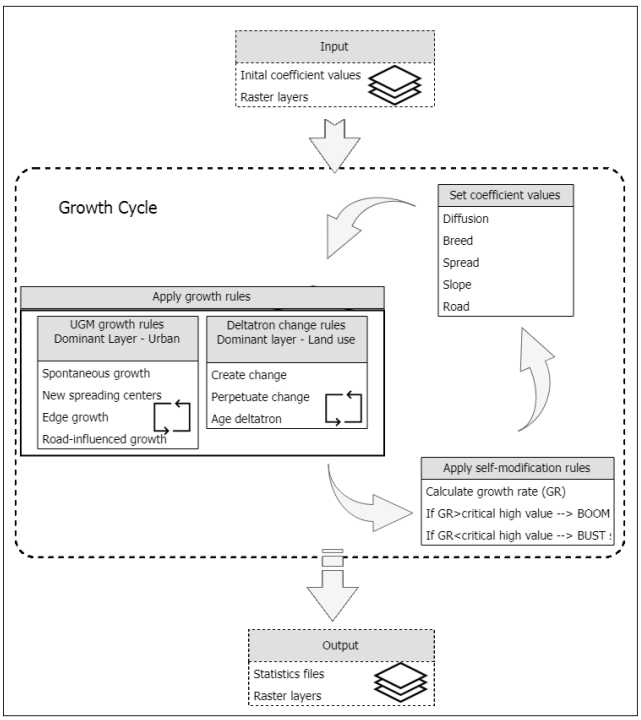

Figure 1: SLEUTH simulation workflow showing growth cycle for each year.

3 DSLEUTH MODIFICATION

SLEUTH is a sound computational model with a long history of accurately calibrating and predicting urban growth. However, the model is computationally expensive due to the large range of parameter values (0-100 inclusive), number of parameters (models – 5), Monte Carlo runs, and time dimensions of the problem. For each combination of parameter values (say 50 for slope, 25 for road, 32 for diffusion, 87 for breed, and 12 for spread), the simulation is run for each temporal unit of input data, times the number of Monte Carlo runs (Guan and Clarke 2010). The amount of time it takes for just one set of parameters on one time for one Monte Carlo run then depends on the size of the image data. In addition to the data size and time period, the calibration time also varies based on amount of urban/non-urban cells in the data space, step-size, number of Monte Carlo iterations, and number of combinations of the five growth parameters. During the calibration stage, a large proportion of the data has to temporarily reside in the virtual memory which greatly increases the overall computing time. Thus calibration is almost impossible for large dataset with high-spatial resolution and long time period on a single processor workstation (Guan and Clarke 2010).

In order to address this time to solution problem, it was observed that evaluating any particular parameter combination is independent of the others and can be executed concurrently. For the purposes of this paper, we will describe a subset of the parameters as a *run*, and the full set of parameters to be explored as a *scenario*. Our goals for splitting the parameter space were:

- Match the number of runs to the number of cores – this would give us good load balancing and keep all of the cores busy.
- Minimize the number of runs – this would give us less overhead. There is overhead associated with reading in the files and setting up the model that we do not want to duplicate more than we need to.

These two goals do not always align and some heuristics were developed to split the parameter space into runs. Additionally, the way the scenario files are specified for SLEUTH limit the ways we can split the space so that we need to be able to specify the run with a single SLEUTH scenario file with contiguous sets of parameters.

Each parameter can be split (each value specified in the scenario file describing the simulation will be in a different run), or not split (all values from the scenario file will be examined in all of the runs). Before the actual set of runs is determined, all possible splits are generated for splitting one, two or three different parameters. For each possible split, we will calculate the number of runs that will be generated if that particular split is chosen. After calculating the number of runs for each way we can split, the splits are scored based on the two criteria above: load balancing on the cores, and reducing the total number of runs. Each case is put into one of the following categories:

1. Fewer runs than number of cores – not viable because it will not use the whole machine.
2. More than ten times the number of cores – not ideal because we want to minimize the number of runs we have due to startup costs.
3. The number of runs is evenly divisible by 0 – this is the ideal load balancing case. Note that the above checks will be evaluated first.
4. Does not fit the other categories but is likely a good candidate.

After each case is evaluated for their score, the best in each category is determined, again seeking to optimize load balancing and minimize number of runs. The best overall split is selected for use by the system and the appropriate scenario files for SLEUTH are generated.

The framework was written in Python and designed to distribute the workload to multiple processes within a shared memory machine with many cores. The framework takes the same scenario file that is used to run SLEUTH, and the number of processes the user wants to run with, as input. The framework then splits up

the work into multiple runs, generates the scenario files for these subsets of the parameter space, launch and manage the executions of SLEUTH such that no more than the specified number of processes are executing SLEUTH at a time. Finally, it will merge the data from the control_stats.log file, containing the information about how well each of the parameter combinations fits the data, into one file so that the user has the same output as she expects from a regular SLEUTH run. One of the goals of this work was to make the user interface familiar and easy to use.

4 DATA

For this study, we used two datasets. The first one was a sample dataset with grid size 200x200, which is available with the SLEUTH3.0beta version of the model (hereby called demo data) (Figure 2). The demo data included four urban layers and transportation layers from 1930, 1950, 1970 and 1990, slope layer, hillshade for visualization and exclusion layer. The exclusion layer showed the waterbodies and protected areas that were not available for urbanization. The second data set was Kolkata data (here by referred as kol data) with grid size of 906*1005 (Figure 3). The kol data was a subset of a dataset produced by another study for the city of Kolkata, India (Chaudhuri and Clarke, under revision). The kol data included four urban layers from 1989, 1999, 2005 and 2010, two road network data from 2004-2005 and 2010, slope layer, hillshade layer for visualization, and exclusion layer demarcating the waterbodies and wetlands. The spatial resolution of the raster images is 100m. In this study, the land use layers were not used because of the mis-match in the number of land use classes over multiple time periods in the kol data. In the SLEUTH model, the land use layer from each time period should have same number of classes. The land use data of kol data from 1989 contains six classes: urban, agricultural, rural, forest, barren and waterbodies. Whereas the land use data from 2010 only contains four classes, due to conversion of forest and barren patches to urban land in core city and suburban areas of Kolkata and to agricultural land in the rural areas by 2010.

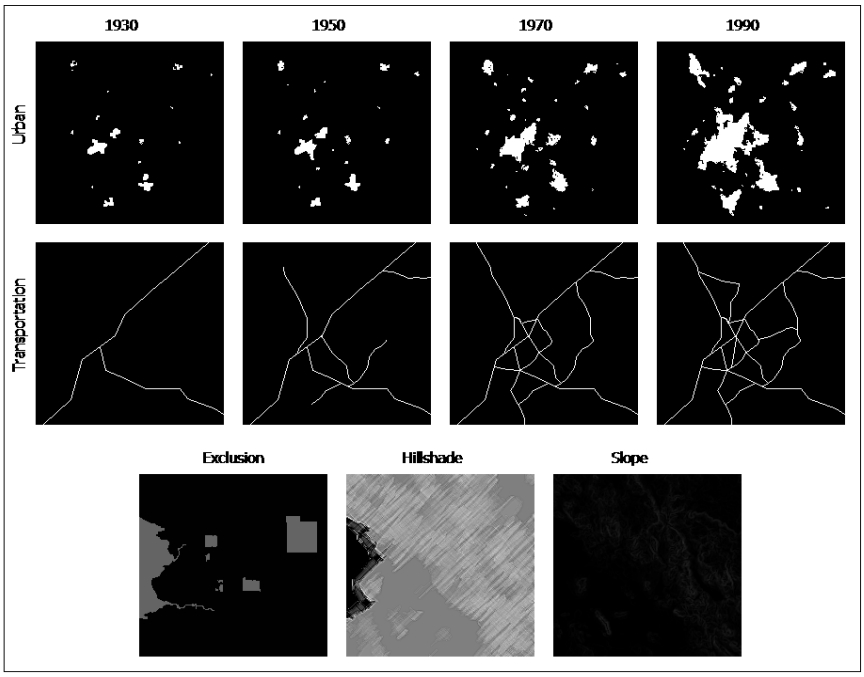

Figure 2: Demo data with grid size 200*200.

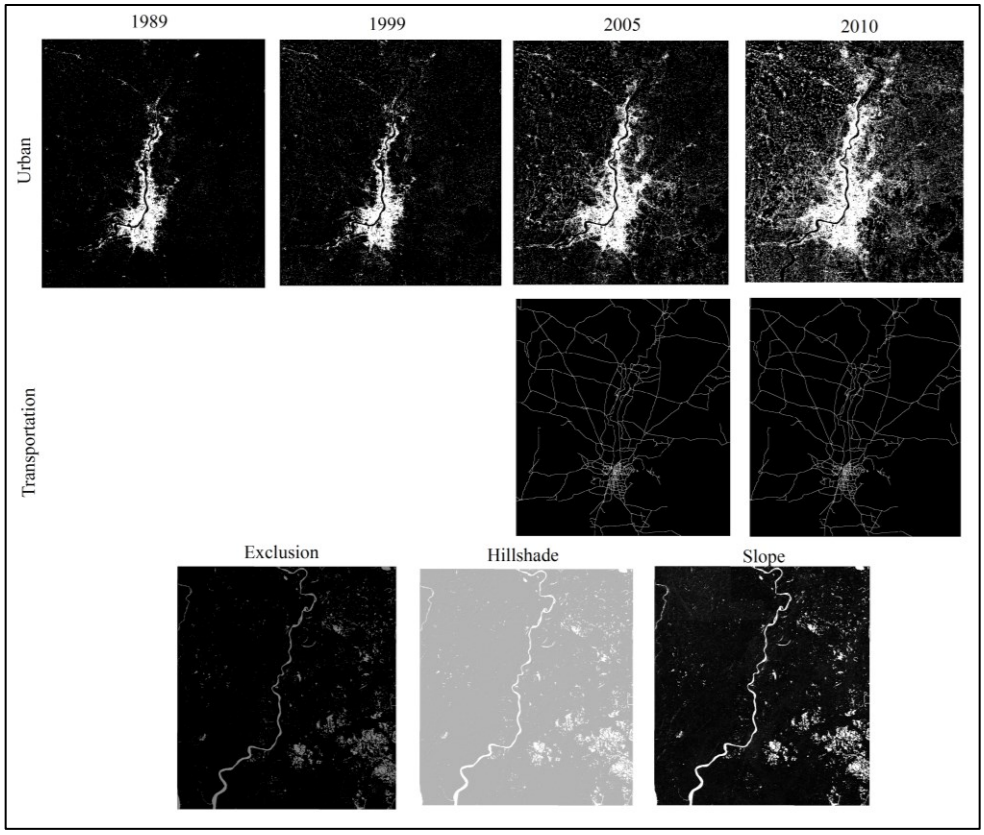

Figure 3: Kolkata data (kol) data with grid size 906*1005.

4.1 Experiment set-up

The SLEUTH and DSLEUTH runs were performed on a Windows 10 workstation with dual processor Intel® Xeon® CPU E5-2640 v4 @ 2.4 GHz machine with 20 cores, 40 virtual processors and 128 GB RAM. The SLEUTH code was run in Cygwin64 and the DLSEUTH version was also run in Cygwin with the DSLEUTH framework running via Python 2.7. We examined several dimensions of the performance of SLEUTH and DSLEUTH to understand how they relate to the time to solution:

- Number of cores via DSLEUTH (1, 10, 20, 30, 40)
- Step size in the first calibration run (10, 20, 25)
- Data size (demo vs. Kolkata)

In this case, the original SLEUTH was used for 1 core and DSLEUTH for multiple cores. Examining the number of cores via DSLEUTH allowed us to answer the question of how memory- versus CPU-bound SLEUTH is, especially with large data sizes. Typically, in SLEUTH the five parameters (Diffusion, Breed, Spread, Slope and Road Gravity) are started with START value 0, STEP value 25 – used to vary the parameter by in successive parameter combinations, and STOP value 100 indicating the upper limit of the parameter. Examining the step sizes 25, 20 and 10 allowed us to examine the tradeoffs of a more accurate initial calibration and if it would allow us to reduce the total number of subsequent calibration runs thus decreasing the overall time to solution. In the subsequent calibration runs the parameter values are narrowed

down guided by the OSM values which shows the best fit parameters. Lastly, the different data sizes allowed us to evaluate if the scaling trends were same for small and large data sets or if they were more dependent on the data themselves.

All calibration runs were performed with 4 Monte Carlo runs. All the log files were flagged to NO except the control_stats.log file (which stores all the parameter values for every Monte Carlo iteration of every run), LOG0 file (which stores the step-by-step model execution, data checking, and transition probabilities) and memory.log file (which stores error messages, if any). For each scenario, we collected the time it took to execute that scenario and monitored the CPU and memory utilization via Windows task manager. Despite the large data size, the memory utilization remained low, even for large numbers of processes.

5 RESULTS: PERFORMANCE COMPARISON OF SLEUTH AND DSLEUTH

Figure 4 shows that DSLEUTH dramatically decreased the time to solution in the first calibration run, however beyond 30 processors the improvements became smaller. The graph (Fig. 4) also shows how a finer grained calibration (step size of 10) versus a larger grained step size (20 or 25) results in more computation being done and a longer time to solution. Figure 5 shows that total time for full calibration was the lowest (best) with the step size 25 for demo data, with only 3 calibration runs. Step sizes 20 and 10 required 4 calibration runs respectively. Step size 10 recorded the longest total calibration time for demo data. For kol data, step size 20 required the least calibration time (best) with 6 runs, whereas step sizes 25 and 10 required almost similar amount of time to calibrate. Figure 5 also shows the OSM values for each step size (higher values suggest better fit). The highest OSM value was recorded for the step size 10 in demo data (0.68868) and step size 20 in kol_data (0.72361). The OSM value was the least for step size 25 for both demo data (0.68102) and kol_data (0.66592). For the demo data, the difference between the best and the worst OSM value was just 1%, but for the kol data its almost 8%. So we can see, for demo data choosing the step size 25 is ideal, whereas for kol data its 20. The total time for calibration using original SLEUTH with 1 core was 6057.12 minutes for demo data, whereas DSLEUTH with 40 cores it was 290.72 minutes for step size 25. For kol data, total time for calibration using original SLEUTH with 1 core was 7090.55 minutes and using DSLEUTH with 40 cores was 372.58 minutes.

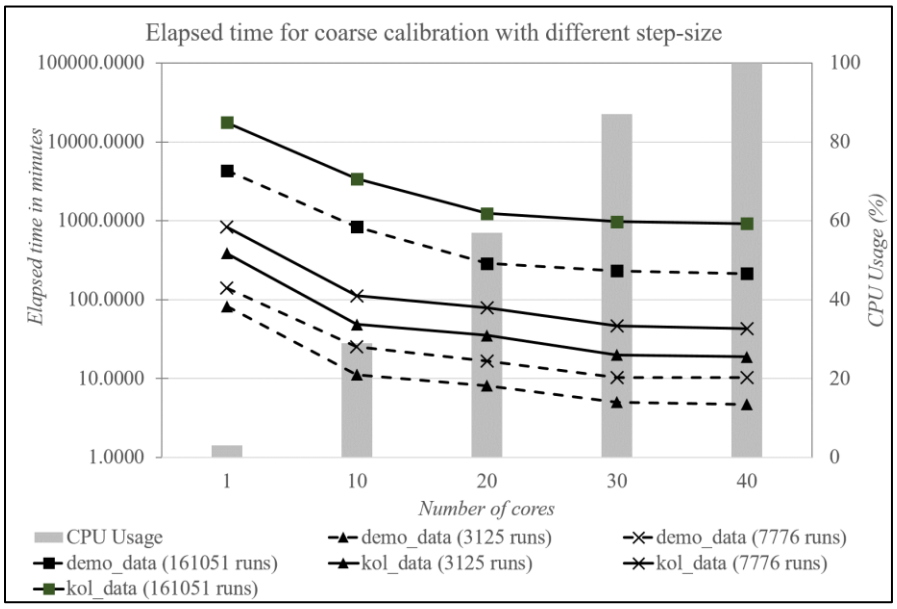

Figure 4: Comparison of elapsed time (in minutes) for coarse calibration using 1, 10, 20 and 40 cores, and corresponding CPU usage for demo data (200*200, dashed lines) and kol data (906*1005, solid lines) using different step sizes.

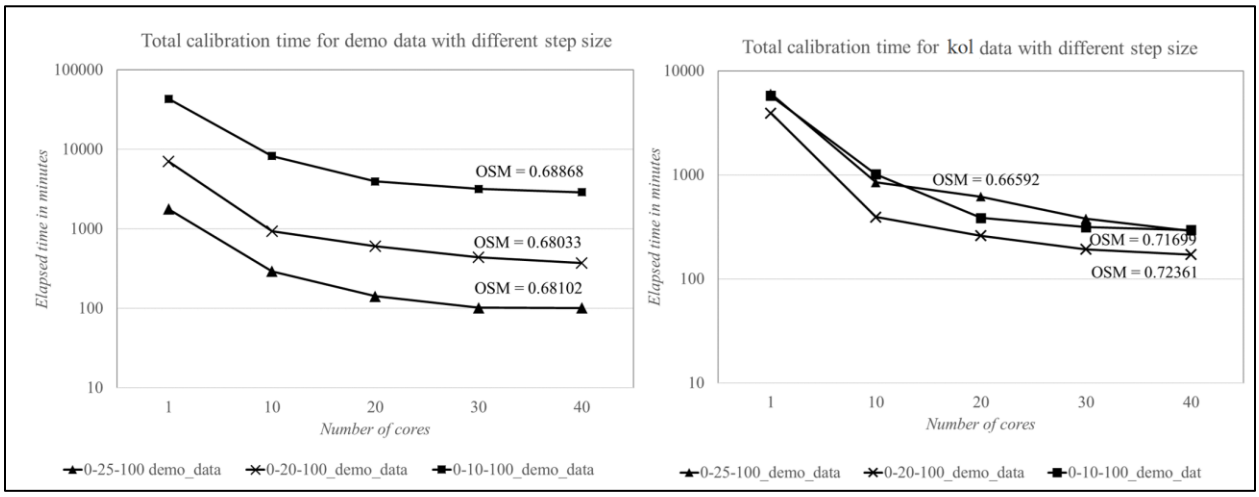

Figure 5: Total time for full calibration of demo data (left) and kol data (right) with different step size and multiple cores.

6 DISCUSSION AND CONCLUSION

The present study used a distributed framework to adapt SLEUTH land use change model, named DSLEUTH, to decrease the calibration time. DSLEUTH allowed modeling larger datasets within a reasonable amount of time without compromising the accuracy of the simulation output. The demo data with 200*200 grid size showed a 21x speed up and kol data with grid size 906*1005 showed a 19x speed up. Accuracy analysis of the predicted kol data from DSLEUTH outside the model showed the exactly same level of accuracy as the predicted output from SLEUTH. Since the demo data is hypothetical in nature,

accuracy analysis was not conducted. The speed up in the demo data provides a good baseline scenario, as it is used by every SLEUTH user to test their model set-up. A current alternative to sequential SLEUTH is the pSLEUTH, a parallel version of the SLEUTH model, which has been proven to decrease the run time and is able to simulate large spatial dataset (Guan and Clarke 2010). pSLEUTH was tested with the kol data. The calibration time for pSLEUTH was lower than DSLEUTH but the accuracy of the predicted output was 60% lower than SLEUTH or DSLEUTH. This is due to pSLEUTH's decomposition of the input data's cell space to distribute the workload (Guan and Clarke 2010). Therefore, although the model ran faster in pSLEUTH, the results were not useful for analysis. On the other hand, DSLEUTH can be used by urban modelers and planners to evaluate real world planning applications with the same accuracy as SLEUTH. This is because the SLEUTH model is used directly and not changed by the framework.

The study showed that the step size of the first calibration run not only affects the run time but also the overall accuracy of the simulation. This is a new information with respect to SLEUTH modeling, as the default step size for first calibration has always been 25 and smaller step sizes were not tested due to long calibration runs. For the demo data, step size 25 yielded the lowest run time and comparable OSM values, whereas kol data step size 20 resulted in the highest OSM value and lowest run time. More testing is required with data from different cities at different level of urbanization to draw a consistent conclusion. However, it is highly likely that the best step size to use may vary from one urban area to another.

Additionally, DSLEUTH needs to be tested with more than 4 Monte Carlo runs. This process will help us to understand whether larger number of Monte Carlo iterations in each calibration run is more time efficient than running more subsequent calibrations with lower number of Monte Carlo iterations. Like the step size, the number of Monte Carlo runs versus the number of subsequent calibration runs need to be tested on multiple urban areas to draw a stable inference.

Finally, DLSEUTH is designed for shared memory machines and can only run on a single machine. In order to truly scale the framework to decrease the runtime further and accommodate even larger datasets, the framework must be modified to work on distributed memory clusters. This has always been our plan and we will extend this work in the near future to work on clusters.

ACKNOWLEDGMENTS

This research was supported by Faculty Research Grant (2015), University of Wisconsin-La Crosse (UWL). A prototype implementation was done as part of the MSE program at UWL by Sheng Zhang and Zhihao Yuan.

REFERENCES

Agarwal, Chetan, Glen M Green, J Morgan Grove, Tom P Evans, and Charles M Schweik. 2002. "A Review and Assessment of Land-Use Change Models: Dynamics of Space, Time, and Human Choice." *General Technical Report NE-297*. Newton Square, PA. http://www.treesearch.fs.fed.us/pubs/5027.

Barredo, JI, and Marjo Kasanko. 2003. "Modelling Dynamic Spatial Processes: Simulation of Urban Future Scenarios through Cellular Automata." *Landscape and Urban Planning*. https://doi.org/10.1016/S0169-2046(02)00218-9.

Batty, M., Yichun Xie, and Zhanli Sun. 1999. "Modeling Urban Dynamics through GIS-Based Cellular Automata." *Computers, Environment and Urban Systems* 23 (3):205–33. https://doi.org/10.1016/S0198-9715(99)00015-0.

Chaudhuri, Gargi, and Keith C. Clarke. 2012. "How Does Land Use Policy Modify Urban Growth? A Case Study of the Italo-Slovenian Border." *Journal of Land Use Science*, no. November:1–23. https://doi.org/10.1080/1747423X.2012.679748.

Chaudhuri, Gargi, and Keith C Clarke. n.d. "Modeling an Indian Megalopolis– a Case Study on Adapting SLEUTH Urban Growth Model." *Computers, Environment and Urban Systems*, no. under revision.

Chaudhuri, Gargi, and Keith C Clarke. 2013. "The SLEUTH Land Use Change Model : A Review." *The International Journal of Environmental Resources Research* 1 (1):88–104.

Clarke, K. C. 2003. "Geocomputation's Future at the Extremes: High Performance Computing and Nanoclients." In *Parallel Computing*, 29:1281–95. North-Holland. https://doi.org/10.1016/j.parco.2003.03.001.

Clarke, K, S Hoppen, and L Gaydos. 1997. "A Self-Modifying Cellular Automaton Model of Historical Urbanization in the San Francisco Bay Area." *Environment & Planning B-Planning & Design* 24:247–61.

Clarke, Keith. 1994. "Project Gigalopolis." Urban Change - Integrated Modeling Environment Webpage. 1994.

Clarke, Keith C. 2017. "Improving SLEUTH Calibration with a Genetic Algorithm." *Proceedings of the 3rd International Conference on Geographical Information Systems Theory, Applications and Management*, no. Gistam:319–26. https://doi.org/10.5220/0006381203190326.

Clarke, Keith C. 2008. "Mapping and Modelling Land Use Change: An Application of the SLEUTH Model." In *Landscape Analysis and Visualisation*, 353–66. Springer.

Dietzel, Charles, and Keith C Clarke. 2007. "Toward Optimal Calibration of the SLEUTH Land Use Change Model." *Transactions in GIS* 11 (1). Wiley Online Library:29–45.

Gazulis, Nicholas, and Keith Clarke. 2006. "Exploring the DNA of Our Regions: Classification of Outputs from the SLEUTH Model." In *7th International Conference on Cellular Automata for Research and Industry*. Perpignan, France. https://doi.org/10.1007/978-3-642-15979-4.

Guan, Qingfeng, and Keith C Clarke. 2010. "A General-Purpose Parallel Raster Processing Programming Library Test Application Using a Geographic Cellular Automata Model." *International Journal of Geographical Information Science* 24 (5). Taylor & Francis:695–722.

Guan, Qingfeng, and Xuan Shi. 2013. "Opportunities and Challenges for Urban Land-Use Change Modeling Using High-Performance Computing." In *Modern Accelerator Technologies for Geographic Information Science*, 227–36. Boston, MA: Springer US. https://doi.org/10.1007/978-1-4614-8745-6_17.

Hawick, Kenneth A, P.D Coddington, and H.A James. 2003. "Distributed Frameworks and Parallel Algorithms for Processing Large-Scale Geographic Data." *Parallel Computing* 29 (10). North-Holland:1297–1333. https://doi.org/10.1016/J.PARCO.2003.04.001.

Hu, Zhiyong, and C.P. Lo. 2007. "Modeling Urban Growth in Atlanta Using Logistic Regression." *Computers, Environment and Urban Systems*. https://doi.org/10.1016/j.compenvurbsys.2006.11.001.

Jantz, Claire, Scott Drzyzga, and Michael Maret. 2014. "Calibrating and Validating a Simulation Model to Identify Drivers of Urban Land Cover Change in the Baltimore, MD Metropolitan Region." *Land* 3 (3):1158–79. https://doi.org/10.3390/land3031158.

Onsted, Jeffrey A., and Rinku Roy Chowdhury. 2014. "Does Zoning Matter? A Comparative Analysis of Landscape Change in Redland, Florida Using Cellular Automata." *Landscape and Urban Planning* 121. Elsevier B.V.:1–18. https://doi.org/10.1016/j.landurbplan.2013.09.007.

Pijanowski, B, D Brown, B Shellito, and G Manik. 2002. "Using Neural Networks and GIS to Forecast Land Use Changes: A Land Transformation Model." *Computers, Environment and Urban Systems* 26 (6):553–75. https://doi.org/10.1016/S0198-9715(01)00015-1.

Pontius, Robert Gilmore, Wideke Boersma, Jean Christophe Castella, Keith Clarke, Ton Nijs, Charles Dietzel, Zengqiang Duan, et al. 2008. "Comparing the Input, Output, and Validation Maps for Several

Models of Land Change." *Annals of Regional Science* 42 (1):11–37. https://doi.org/10.1007/s00168-007-0138-2.

Rienow, Andreas, and Roland Goetzke. 2015. "Supporting SLEUTH - Enhancing a Cellular Automaton with Support Vector Machines for Urban Growth Modeling." *Computers, Environment and Urban Systems* 49. Elsevier Ltd:66–81. https://doi.org/10.1016/j.compenvurbsys.2014.05.001.

Shashidharan, Ashwin, Derek B. van Berkel, Ranga Raju Vatsavai, and Ross K. Meentemeyer. 2016. "PFUTURES: A Parallel Framework for Cellular Automaton Based Urban Growth Models." *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9927 LNCS:163–77. https://doi.org/10.1007/978-3-319-45738-3_11.

Silva, Elizabete A, and Keith C Clarke. 2002. "Calibration of the SLEUTH Urban Growth Model for Lisbon and Porto, Portugal." *Computers, Environment and Urban Systems* 26 (6). Elsevier:525–52.

Solecki, William D, and Charles Oliveri. 2004. "Downscaling Climate Change Scenarios in an Urban Land Use Change Model." *Journal of Environmental Management* 72 (1–2):105–15. https://doi.org/http://dx.doi.org/10.1016/j.jenvman.2004.03.014.

Tayyebi, Amin, Burak K Pekin, Bryan C Pijanowski, James D Plourde, Jarrod S Doucette, and David Braun. 2013. "Hierarchical Modeling of Urban Growth across the Conterminous USA: Developing Meso-Scale Quantity Drivers for the Land Transformation Model." *Journal of Land Use Science* 8 (4):422–42. https://doi.org/10.1080/1747423x.2012.675364.

Torrens, Paul M, and David O'Sullivan. 2001. "Cellular Automata and Urban Simulation: Where Do We Go from Here?" *Environment and Planning B: Planning and Design*. https://doi.org/10.1068/b2802ed.

AUTHOR BIOGRAPHIES

GARGI CHAUDHURI is an Associate Professor of Geography and Earth Science at University of Wisconsin-La Crosse. She holds a PhD in Geography from University of California Santa Barbara. Her research interests lie in geographic information science, urbanization and simulation modeling. Her email address is gchaudhuri@uwlax.edu.

SAMANTHA FOLEY is an Assistant Professor of Computer Science at University of Wisconsin-La Crosse. She holds a Ph.D. in Computer Science from Indiana University. Her research interests include high performance computing, scientific computing, and computer science education at the K-12 level, focusing on helping domain scientists leverage the power of parallel and distributed systems to solve problems more quickly and easily. Her email address is sfoley@uwlax.edu.