Training Cellular Automata to Simulate Urban Dynamics: A Computational Study Based on GPGPU and Swarm Intelligence

Ivan Blecic, Arnaldo Cecchini, and Giuseppe A. Trunfio

Department of Architecture, Planning and Design, University of Sassari, Italy
{ivan,cecchini,trunfio}@uniss.it

Abstract. We present some results of a computational study aimed at investigating the relationship between the spatio-temporal data used in the calibration phase and the consequent predictive ability of a Cellular Automata (CA) model. Our experiments concern a CA model for the simulation of urban dynamics which is typically used for predicting spatial scenarios of land-use. Since the model depends on a large number of parameters, we calibrate the CA using Cooperative Coevolutionary Particle Swarms, which is an effective approach for large-scale optimizations. Moreover, to cope with the relevant computational cost related to the high number of CA simulations required by our study, we exploits the computing power of Graphics Processing Units.

Keywords: Cellular Automata, Urban Models, Cooperative Coevolution, GPU.

1 Introduction

Cellular Automata (CA) models for the simulation of urban dynamics can provide to decision-makers and urban planners accurate information on the growth of cities, including the types, location and amount of land-use conversions . For this reason, they have been used in several ways to support land use planning, policy evaluation, and analysis of future scenarios of urban growth [1–3]

Among the open research challenges in the field of CA-based modelling of urban systems, a recurrent issue is represented by *calibration*, which consists of adapting the parameter-dependent transition rules to make the modelled urban phenomena matching the reality. CA calibration is often challenging because it involves high-dimensional search spaces. To cope with this problem, the current research trend in the field suggests the use of automated optimisation procedures [4–6]. The latter, are based on the availability of historical maps of the area under study, which are used, together with an appropriate metric of agreement, to guide the search of the values of model parameters. In this regard, an important issue concerns the relationship between the predictive ability of the calibrated model and the historical maps used in the optimization process. For example, for a CA modeller is relevant to be aware of the possible calibration error, with the related

J. Wąs, G.C. Sirakoulis, and S. Bandini (Eds.): ACRI 2014, LNCS 8751, pp. 300–309, 2014.

loss of model accuracy, that can be expected when very few historical maps are available to support the optimization process. To our knowledge, this issue was not addressed by systematic studies in the literature.

In this paper we present some results of a broader computational study on the calibration of a constrained CA for simulating urban dynamics. Our main objective was to investigate the extent to which the availability of historical maps and the way in which they are used, may affect the quality of the calibration process, as well as the predictive power of the resulting model. Our experiments take advantage of a parallel CA model already presented in [7], which is based on general-purpose computing on graphics processing units (GPGPU). This allowed us to perform an empirical investigation based on the execution of several millions of CA simulations in a reasonable time. Moreover, in order to cope with the high number of parameters involved in the calibration, we use a state-of-the-art metaheuristic, namely the Cooperative Coevolutionary Particle Swarm optimization (CCPSO) [8], which is a variation of the standard Particle Swarm Optimization (PSO) algorithm [9] specifically designed to deal with optimizations in spaces with a high number of dimensions.

2 The CA Urban Model and Its Calibration Problem

We use a CA representing the geographical space of interest and evolved in order to mimic its land-use dynamics over time. The model was parallelized for the GPGPU platform provided by nVidia, as explained in [7]. In order to obtain high efficiency, we formulated the parallelization so as to avoid significant memory transfers between the CPU and GPU during the simulation. As a result, our GPGPU approach leads to speedups that can easily exceed the value of 100, compared to the corresponding sequential implementation on a standard workstation [7]. This was key factor to allow the presented investigation.

In our CA model, the relevant component of the state of each cell represents its land-use class (such as residential, industrial, commercial, agriculture). Cells may also hold other information relevant to the simulation, such as their distance from the main transportation networks, constraints related to zoning regulations and cells' physical features (slope, elevation, etc.). During the simulation of such a CA, each cell can change its land use depending on its neighbouring cells and its internal state. However, the cell's state transition also depends on some global constraints on the total amount of each land use that is allowed at each time step. The model, as proposed in [1], includes three categories of land uses: *(i) static*, which cannot change during the simulation (e.g. transportation network, public services and facilities). A static land-use can however influence the other uses within its neighbourhood by exerting attractive or repulsive effects on them; *(ii) active* uses, for which there is an explicit demand (e.g. in terms of area) at each time step; *(iii) passive* uses, representing land available to be transformed into active uses during the simulation.

In the adopted model, the CA neighbourhood is defined as the square region around the cell with a sufficient size to allow local-scale spatial processes to be

captured in the CA transition rules. At the beginning of each CA step, the so-called transition potentials P_j [1, 10] are computed for each cell and each active land use. This is done with the following equation:

$$P_j = I_i + \gamma\, S_j\, Z_j\, N_j \tag{1}$$

where:

- i is the current cells land use and $I_i \geq 0$ represents an inertia due to the transformation costs of transition from the use i to a different use;
- $\gamma_j = 1 + (-\ln \psi)^{\alpha_j}$, where ψ is a random number between 0 and 1 and α_j is a parameter that provides a degree of randomness;
- $S_j \in [0, 1]$ is the *suitability* factor for the active land use j, which is expressed as a logistic function of n_f local predictors x_k (e.g. the cell's distance from the street network or the terrain slope) as follows:

$$S_j = \frac{exp(\sum_{k=1}^{n_f} b_{kj} x_k)}{1 + exp(\sum_{k=1}^{n_f} b_{kj} x_k)} \tag{2}$$

  where the parameters b_{kj} are estimated through the calibration process.
- $Z_j \in [0, 1]$ defines the degree of legal or planning permissibility of the j-th land use (for example due to zoning regulations by the planning authority);
- N_j is the so called *neighbourhood effect* computed as:

$$N_j = \nu + \sum_{c \in V} \phi(a_{i,j},\, b_{i,j},\, c_{i,j},\, d_{i,j};\, \delta_c) \tag{3}$$

  where the summation is extended over all the cells of the cell's neighbourhood V and: i denotes the current land use of the cell $c \in V$, δ_c is the distance from the neighbouring cell c, and $\phi(a_{i,j},\, b_{i,j},\, c_{i,j},\, d_{i,j};\, \delta)$ is a piecewise linear function, depending on the four scalar parameters, $a_{i,j}$, $b_{i,j}$, $c_{i,j}$ and expressing the influence of the i-th land-use at the distance δ on the potential land use j. The term ν is a constant value computed before the beginning of the simulation so that $N_j \geq 0$.

Once the cell's P_j have been computed, the CA evolution is consists of transforming each cell into the state with the highest potential, given the exogenous constraint on the overall number of cells in each state imposed for that step [1].

The dynamics of the CA model described above depends on many scalar parameters that must be adapted to the specific application context. In particular, if the model includes n_s static land uses, n_p passive uses and n_a actively modelled uses, it depends on: *(i)* n_a parameters I_i defining the inertial contribution to the transition potentials; *(ii)* $n_f\, n_a$ parameters involved in the logistic suitability defined by Eq. (2); *(iii)* $4\,(n_s + n_a + n_p)\, n_a$ parameters a_{ij}, b_{ij}, c_{ij}, d_{ij} involved in the piecewise functions $\phi_{i,j}$ of Eq. (3); *(iv)* n_a parameters α_j defining the degree of randomness. All the above parameters can be collected in a vector $\mathbf{p}$ belonging to a D-dimensional search space. With respect to $\mathbf{p}$, the model can be optimised to maximise the fitting between the simulated and real patterns.

In the following we indicate with $\bar{\mathcal{V}}$ a *training set* collecting some maps $\bar{\omega}^{(t)}$, which correspond to historical land-use data on the area under study. Starting from a known configuration $\omega^{(0)}$, and given a vector $\mathbf{p}$ of parameters, the CA can be executed for the computation of a set $\mathcal{V}$ of automaton configurations $\omega^{(t)}$ corresponding to the training maps $\bar{\omega}^{(t)}$.

The agreement between the real spatio-temporal sequence and the simulated one should be quantified through a suitable measures of fitness $\Theta\left(\bar{\mathcal{V}}, \mathcal{V}\right)$, which is computed for each value of $\mathbf{p}$. Given the fitness function, in our view the automatic calibration consists of maximising $\Theta\left(\bar{\mathcal{V}}, \mathcal{V}\right)$, with respect to $\mathbf{p}$, through a suitable search algorithm. To this purpose, we use a variation of the standard Particle Swarm Optimization (PSO) algorithm [9], which was specifically designed to deal with optimizations in spaces with a high number of dimensions, namely the Cooperative Coevolutionary PSO (CCPSO) approach [8]. The latter was already successfully tested for urban CA calibrations in [11], where the details on its formalization and implementation can be found.

3 Computational Study

3.1 Experimental Setup

We applied the model to the area of the city of Heraklion, Crete. The CA representing the urban area was composed of 277×151 cells each with the side of 50 meters and the initial configuration was initialized with the urbanization map of the year 1980, labelled as $\omega^{(0)}$ in Fig. 1. In the model, we included seven land uses, four of which actively modelled. The latter were *residential dense*, *residential sparse*, *industrial areas*, and *commercial areas*. The *undeveloped land*, which essentially represents agricultural and natural land cover classes, was considered as the only passive land use. The only static land use was *green urban areas and facilities*. Also, we used a square neighbourhood with side of 20 cells. For determining the suitabilities given by Eq. (2) we used six driving factors, namely: *distance from main roads*, *distance from secondary roads*, *terrain slope*, *altitude*, *distance from the sea* and *distance from the city center*. In order to reduce the number of unknown model parameters, we adopted the value of 0.01 for all the α_i that define the simulation randomness. Given the above characteristics, the model depends on $D = 140$ unknown parameters.

To measure the agreement between maps during calibration, we use a modified version of the standard Kappa statistic, namely the so called Kappa Simulation K_s [12]. The standard Kappa measures the agreement between two categorical datasets relative to the expected agreement by chance (i.e. when the given sizes of classes are reallocated randomly). In the modified K_s version, the agreement between the two maps is corrected accounting for the sizes of class transitions, which are computed taking as a reference the initial map. We defined the fitness function as follows:

$$\Theta\left(\bar{\mathcal{V}}, \mathcal{V}\right) = \frac{1}{|\bar{\mathcal{V}}|} \sum_{\bar{\mathcal{V}}, \mathcal{V}} K_s(\omega^{(0)}, \bar{\omega}^{(\tau_i)}, \omega^{(\tau_i)}) \qquad \bar{\omega}^{(\tau_i)} \in \bar{\mathcal{V}},\ \omega^{(\tau_i)} \in \mathcal{V} \qquad (4)$$

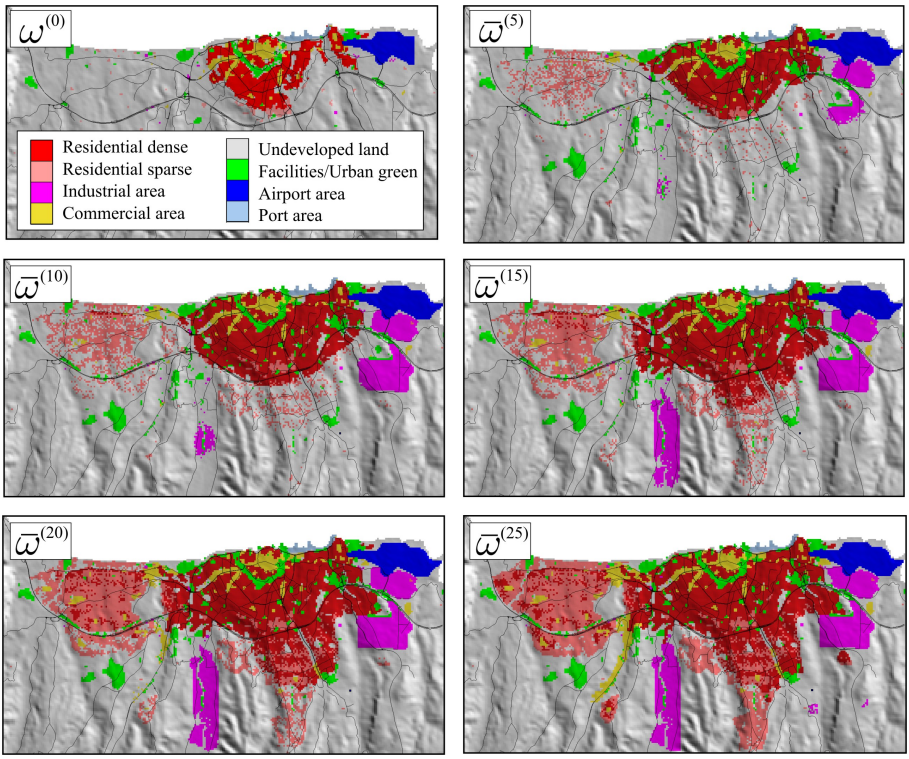

Fig. 1. Some CA configurations produced through the randomly drawn model parameters and used in the calibration study

where the τ_i indicates the time step in which the configurations are known. In other words, we defined the fitness as the arithmetic mean of all the K_s computed on the corresponding training and simulated maps.

In a preliminary stage of our numerical experimentation, we produced a training set through the CA itself. This guaranteed the existence of a zero-error solution of the calibration problem, thus allowing for an unbiased evaluation of the results obtained through the calibration procedures. To such purpose, we randomly generated the 140 unknown parameters of the model within plausible ranges, trying to achieve a fairly realistic land-use dynamics. Using such parameters, the CA simulation was then performed for 25 steps obtaining the dynamics shown in Fig. 1 and a set of CA configurations

$$\bar{\mathcal{V}} = \left\{ \omega^{(t)} \mid t = 1 \ldots 25 \right\} \tag{5}$$

It is important to note that for our purposes would not be useful to derive a vector of parameters by means of a calibration process based on a real map. Indeed, this would introduce a bias in the sense that new calibrations would tend to provide vectors of parameters that better reproduce the configurations used

for the original calibration. This does not apply if the parameters are randomly drawn, though within predetermined intervals.

The computational experiments discussed in the following essentially consist of using training sets composed of some of the maps in $\bar{\mathcal{V}}$ with the aim of reproducing the correct dynamics after model calibration.

As for the constraints, for the 15th CA step we used the actual amount of land for each land-use taken from a 2010 map of the area. Then, we adopted a scenario of linear increment in land-use demand between steps 0 and 25.

We assigned to each calibration a budget of 20000 CA evaluations and we carried out 10 independent runs for each experiment, averaging the results in terms of achieved fitness. Moreover, we ran the optimization algorithms on a workstation equipped with two different GPUs: the nVidia Tesla K40 and a nVidia Geforce GTX 680 graphic card. In order to exploit both GPUs, we developed a multi-GPU program using the C++/CUDA languages and a multi-threads approach according to a master-slaves paradigm.

3.2 Results and Discussion

In the study we considered a number of different training sets composed of one or more maps taken from the set $\bar{\mathcal{V}}$ defined in Eq. 5. However, due to space limitations only some results will be discussed here. In the following, we indicate with $\bar{\mathcal{V}}_i$ a training set including only $\bar{\omega}^{(i)}$, with $\bar{\mathcal{V}}_{i,j}$ a training set including $\bar{\omega}^{(i)}$, $\bar{\omega}^{(j)}$ and so on. In Fig. 2 we show the averaged convergence plots of the optimizations for some experiments. The statistics on the attained fitnesses are shown in Table 1.

As expected, a calibration point after only five CA steps gave rise to a relatively easy optimization problem. This is due to the small differences between the two configurations $\omega^{(0)}$ and $\bar{\omega}^{(5)}$. More in details, according to Table 1 in this case the optimization algorithm was able to achieve the average fitness $\Theta = 0.980$. Note that because of the probabilistic nature of the optimization algorithm, given a training set, a lower standard deviation implies a lower number of optimization runs that are needed to obtain a reliable calibration. As shown by Fig. 2 and Table 1, including calibration points in more advanced time steps, produced a reduction of the speed of convergence with decreased fitness value and increase of the variability of the calibration results. In most calibration configurations a t-test on the results showed that the achieved fitness values were essentially equivalent. Note however that such an equivalence only refers to the achieved fitness considered as a casual variable given by a probabilistic calibration process. As shown later, the used training sets can indeed lead to very different simulation qualities.

Table 1 also shows the average computation time which was necessary for an optimization: Thanks to the GPGPU acceleration, the average time took by a 25-steps CA simulation was 0.2 s.

Our experimental setup allows to examine the quality of calibration with respect to all reference configurations included in $\bar{\mathcal{V}}$ (see Eq. 5). To this end, we used the best parameter vectors obtained in the calibrations and we ran the

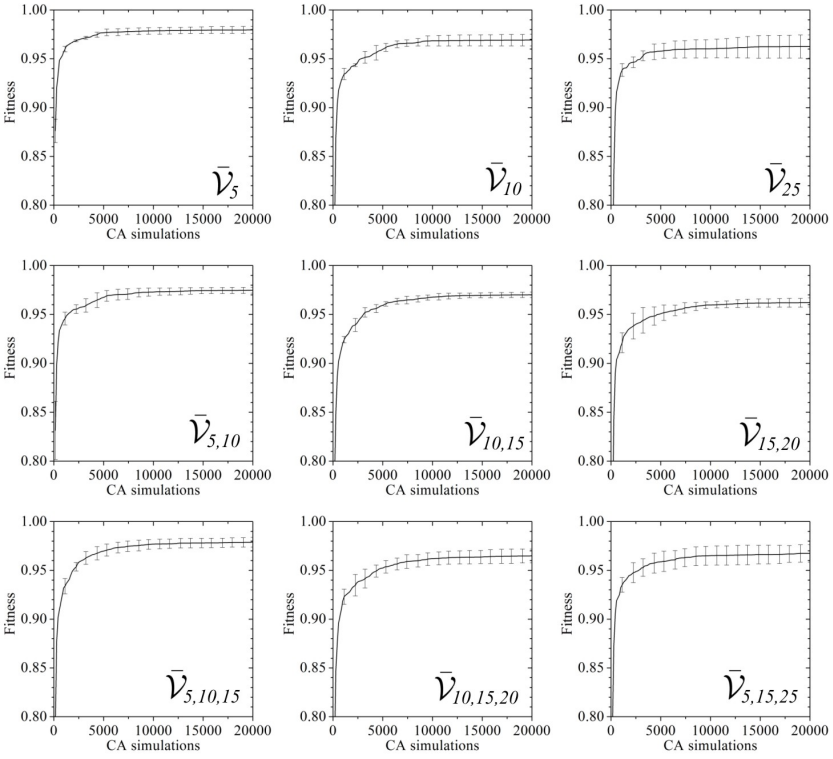

Fig. 2. Convergence plots of the average fitness Θ and its standard deviation for some calibration tests. The statistics were computed over 10 independent runs on each test.

Table 1. Statistics on the achieved fitness Θ and elapsed time in seconds over 10 independent runs for some calibration tests

Training set	$\bar{\mathcal{V}}_5$	$\bar{\mathcal{V}}_{10}$	$\bar{\mathcal{V}}_{25}$	$\bar{\mathcal{V}}_{5,10}$	$\bar{\mathcal{V}}_{10,15}$	$\bar{\mathcal{V}}_{15,20}$	$\bar{\mathcal{V}}_{5,10,15}$	$\bar{\mathcal{V}}_{10,15,20}$	$\bar{\mathcal{V}}_{5,15,25}$
Avg	0.980	0.969	0.963	0.975	0.970	0.962	0.979	0.965	0.967
Std. Dev.	0.004	0.006	0.012	0.003	0.003	0.005	0.005	0.007	0.009
Min	0.975	0.964	0.952	0.970	0.967	0.955	0.974	0.955	0.956
Max	0.983	0.977	0.975	0.977	0.973	0.966	0.984	0.972	0.976
Avg. Time	1171	2099	4647	2104	3305	3686	3007	3772	4763

corresponding CA simulation by calculating the adopted measure of agreement K_s between $\bar{\omega}^{(t)}$ and $\omega^{(t)}$ at each step t. The objective was to obtain a quantification of the accuracy that can be achieved along the simulation. Some validation results are summarized in Table 2. It is also interesting to examine in details the values of K_s along the simulation for the different training sets, which are depicted in Fig. 3. According to the results, the actual quality of calibrations in the

Table 2. Validation results for some of the calibration tests. The statistics were computed on the validation set (composed of 25 maps) in terms of K_s.

	$\bar{\mathcal{V}}_5$	$\bar{\mathcal{V}}_{10}$	$\bar{\mathcal{V}}_{15}$	$\bar{\mathcal{V}}_{5,10}$	$\bar{\mathcal{V}}_{5,15}$	$\bar{\mathcal{V}}_{10,15}$	$\bar{\mathcal{V}}_{15,20}$	$\bar{\mathcal{V}}_{5,10,15}$	$\bar{\mathcal{V}}_{10,15,20}$
Average K_s	0.958	0.971	0.960	0.970	0.983	0.971	0.967	0.983	0.972
Worst K_s	0.934	0.957	0.952	0.959	0.977	0.957	0.962	0.972	0.967

considered range of 25 CA steps, was significantly different for the used training sets. In general, there was a good K_s in a neighbourhood of the calibration point (note that in correspondence to the latter, for the single point calibration, the K_s value corresponds to the fitness Θ achieved during calibration). However, the deterioration of the simulation quality greatly depended on where the calibration occurred. For example, from Fig. 3 it is clear that the easiest calibration at the step 5, in spite of the high value of the achieved fitness (i.e. $\Theta = 0.980$), did not provide a suitable vector of model parameters, leading to the low average agreement of 0.958 in terms of K_s (see Table 2). This is not surprising, given that the map produced by a single CA step does not contain enough information on the system's dynamics. Also, looking at the other extreme, the calibration at the farthest point from the origin (i.e. using $\bar{\mathcal{V}}_{25}$) produced a good agreement only in the immediate vicinity of the initial and final configurations, with an average of 0.957. However, with a single calibration-point in an intermediate position (e.g. for $\bar{\mathcal{V}}_{10}$ and $\bar{\mathcal{V}}_{15}$) the evolution of agreement during the simulation showed a low variability and an average value quite close to that obtained in the calibration phase. Interestingly, in our case study, we can achieve a reasonable quality of the calibrated model using a single historical map, as long as it does not refer to a point which is too close or too far from the beginning of the simulation.

Better results were obtained using more than one calibration point. In particular, using $\bar{\mathcal{V}}_{5,15}$ led to a good agreement with the validation set along the whole simulation, characterized by an average K_s of 0.983 with a minimum of 0.977. According to the results in Fig. 3, it seems that using only two calibration points, provided they are well spaced and not too close to the beginning of the simulation, can lead to a suitable CA model optimization. The validation of calibration based on the training set $\bar{\mathcal{V}}_{5,10,15}$ showed a quite stable K_s during the simulation, with an average value of 0.983 and a worst value of 0.972. However, these value are substantially equivalent to those obtained using the smaller training set $\bar{\mathcal{V}}_{5,15}$ (see Table 2). Therefore, increasing the training set $\bar{\mathcal{V}}_{5,15}$ with an intermediate map proved essentially useless. Clearly, this is because, in our case study, the map $\bar{\omega}^{(10)}$ did not provide any supplementary informational contribution. It is important to highlight that the insights that can be drawn from the experiments described above refer to a case in which the rules that determine the urban dynamics (i.e. the parameters) are constant over time. However, this can be, at least approximately, the case for many historical periods and urban areas in the world.

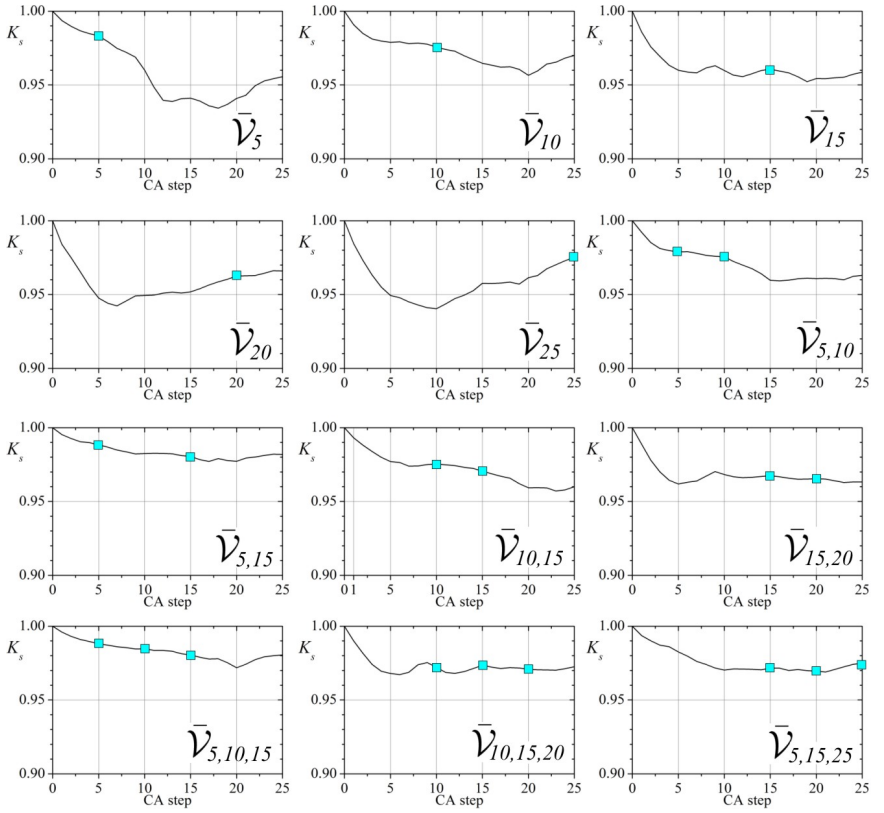

Fig. 3. Agreement, in terms of K_s, with the validation set during the simulation. Calibration points are highlighted.

The results can also be examined from a forecasting perspective. For example, supposing that some maps corresponding to CA steps between 0 and 15 (i.e. the hypothetical current time) are available, we can estimate the influence of calibration error in forecasting the CA configuration at the 25th step (the hypothetical future time). As can be seen from Fig. 3, the best forecasting ability was achieved through a model calibrated using $\bar{\mathcal{V}}_{5,15}$ (final K_s of 0.982) and $\bar{\mathcal{V}}_{5,10,15}$ (final K_s of 0.980). Using a single calibration point led to poor predictive performance of the model: the values of K_s at the 25th CA step were, 0.956, 0.970 and 0.959 for $\bar{\mathcal{V}}_5$, $\bar{\mathcal{V}}_{10}$ and $\bar{\mathcal{V}}_{15}$, respectively.

4 Conclusions and Future Work

Constrained CA for land use change simulations often depends on many parameters that must be determined through a calibration process. The latter usually requires the availability of a training set containing an adequate number of historical maps of the area under study. Using a suitable experimental setup

and some advanced computational techniques, we investigated the influence of the composition of the training set on the quality of the resulting calibration. We presented and discussed in this paper only some experiments. However, the whole study required the execution of more than 5 million simulations. In return, it provided numerous insights and allowed us to quantify in various conditions the calibration and validation errors, as well as their implications in the case of land-use change forecasting. Future work will investigate the effect of the different training sets using a multi-objective metaheuristics [13], in which the fitness function is based on various landscape metrics.

References

1. White, R., Engelen, G.: High-resolution integrated modelling of the spatial dynamics of urban and regional systems. Computer, Environment and Urban Systems 24, 383–400 (2000)
2. Barredo, J.I., Kasanko, M., McCormick, N., Lavalle, C.: Modelling dynamic spatial processes: simulation of urban future scenarios through cellular automata. Landscape and Urban Planning 64, 145–160 (2003)
3. Blecic, I., Cecchini, A., Falk, M., Marras, S., Pyles, D.R., Spano, D., Trunfio, G.A.: Urban metabolism and climate change: A planning support system. Int. J. Applied Earth Observation and Geoinformation 26, 447–457 (2014)
4. Feng, Y., Liu, Y., Tong, X., Liu, M., Deng, S.: Modeling dynamic urban growth using cellular automata and particle swarm optimization rules. Landscape and Urban Planning 102, 188–196 (2011)
5. Rabbani, A., Aghababaee, H., Rajabi, M.A.: Modeling dynamic urban growth using hybrid cellular automata and particle swarm optimization. Journal of Applied Remote Sensing 6 (2012)
6. Blecic, I., Cecchini, A., Trunfio, G.A.: A comparison of evolutionary algorithms for automatic calibration of constrained cellular automata. In: Taniar, D., Gervasi, O., Murgante, B., Pardede, E., Apduhan, B.O. (eds.) ICCSA 2010, Part I. LNCS, vol. 6016, pp. 166–181. Springer, Heidelberg (2010)
7. Blecic, I., Cecchini, A., Trunfio, G.A.: Cellular automata simulation of urban dynamics through GPGPU. The Journal of Supercomputing 65, 614–629 (2013)
8. van den Bergh, F., Engelbrecht, A.P.: A cooperative approach to particle swarm optimization. IEEE Trans. Evolutionary Computation 8, 225–239 (2004)
9. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995)
10. Wu, F.: SimLand: A prototype to simulate land conversion through the integrated GIS and ca with ahp-derived transition rules. International Journal of Geographical Information Science 12, 63–82 (1998)
11. Blecic, I., Cecchini, A., Trunfio, G.A.: Fast and accurate optimization of a GPU-accelerated CA urban model through cooperative coevolutionary particle swarms. Procedia Computer Science 29, 1631–1643 (2014)
12. van Vliet, J., Bregt, A.K., Hagen-Zanker, A.: Revisiting Kappa to account for change in the accuracy assessment of land-use change models. Ecological Modelling 222, 1367–1375 (2011)
13. Blecic, I., Cecchini, A., Trunfio, G.A.: A decision support tool coupling a causal model and a multi-objective genetic algorithm. Appl. Intell. 26, 125–137 (2007)