Calibrating probabilistic cellular automata for agent-based modelling of real systems

Pranjal Dhole[1], Alexander Asteroth[1] and Stefanie Meilinger[1]

[1]Hochschule Bonn-Rhein-Sieg, Sankt Augustin, Germany 53177
pranjal.dhole@inf.h-brs.de

Mathematical models are simplifications of the real systems. The value of a mathematical model lies in its ability to reasonably model real-world phenomena. Cellular automata (CA) are a sub-class of agent-based models (ABM) that have been quite successful in modelling interacting real systems such as traffic flows, crowd behaviour, ecology, epidemics, chemical systems, etc. However, in the literature, CA models have been criticized as oversimplifications of reality and those that have relaxed rules have been criticized as not pure CA. In this paper, we show a procedure for calibrating a probabilistic-CA (P-CA) model to real system.

CA are discrete in space and in time which makes them ideal for high performance computer simulations. The time evolution of CA depends on initial-value-conditions and boundary-value conditions of the simulation. The introduction of probabilistic state update rules in P-CA made the initial-value conditions in P-CA irrelevant. The focus in P-CA models lie in the generation of consistent macroscopic distributions irrespective of initial-value-conditions. Consequently, not much effort has been put in calibration of P-CA to model real systems. In this paper, we investigate a simple scenario of open system and perform calibration of P-CA model such that the model can be validated as a model for real-world system.

Model verification, calibration and validation are important steps in the development of a credible simulation. Model verification implies building the model correctly, i.e., we need to accurately transform the model concept from simulation flowchart into a computer program. Model calibration is the process of obtaining a desired confidence level on the model such that its results are considered reasonable for the objective it was developed for. This involves selection of right model parameters (C_{sim}) and the right input data for the simulation (I_{sim}). Model validation is the process of making sure that the output data obtained from the simulation driven by input data are close to real-system output data. When comparing the system and the model output data, if there are substantial differences in the comparison, some correction factors are added to the model input data. Thereafter, the model and the system are compared again.

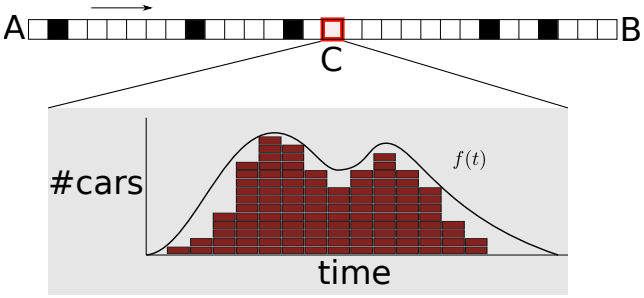

Figure 1: Traffic flow counts measured at a cell (red) when n cars are running on a single lane over a simulation time T.

This iterative process of input modification to meet simulation output that is close to real system output is called calibration.

In this paper, we consider a one-dimensional P-CA model called Nagel-Schreckenberg (NaSch-)model (Nagel and Schreckenberg, 1992). NaSch-model is a regarded as a prototype CA-model for modelling traffic flows with the smallest set of state update rules: (i) acceleration, (ii) deceleration due to other cars, (iii) random deceleration to model variability in driving behaviour and (iv) position update.

Fig. [1] shows a one-dimensional lane where vehicles move from point A to point B. This is the simplest example of real-world traffic scenario. The model parameters to calibrate in the NaSch-model are maximum limiting speed ($v_{\max}$), safety distance (d) and state transition probability (also known as breaking parameter, p). We synthetically generate ground-truth data by selecting model parameters and a vehicle initialization scheme Φ_A at point A. Φ_A is fed to the simulation as input data. We consider the total number of cars $n = 92$ running on a lane consisting of $L = 100$ cells with total simulation runtime $T = 600$. The simulation time is discretized into 20 bins such that $w = 30$ time steps are used to aggregate vehicle count in each bin.

The ground-truth input data Φ_A^* is generated by sampling from two non-normalized gaussian distributions $\mathcal{N}(8, 2)$ and $\mathcal{N}(14, 3)$ as shown in Fig. [2]. In the calibration procedure, we need to calibrate both model parameters as well

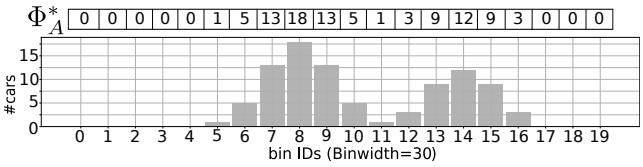

Figure 2: Synthetically generated ground-truth input data (Φ^*_A) for the simulation.

as the vehicle initialization scheme Φ_A at point A such that the distribution of the simulation traffic-count at point C matches with the ground truth distribution of the simulation traffic-count.

For the purposes of calibration, we use genetic algorithms (GA) which has proven to be a successful stochastic search method for evolving CA. The algorithmic loop for GA is shown in Fig. [3]. GA employs a set of genetic operations such as mutation, crossover, etc. for generating new set of input data for the simulation. For a tutorial on the application of GA in practice, we refer to (Asteroth and Hagg, 2015).

In our experiment, we define a discrete-valued genome which is a vector of traffic counts. This 1×20 vector defines the vehicle initialization scheme at point A. The values in the genome define the number of cars that are initialized within that time window during the simulation run. The upper bound for each gene in genome is set to the time-interval within one time bin ($w = 30$). This upper-bound ensures that we do not have multiple cars in the same cell at the same time thereby keeping the model collision-free. We separate the car initialization time within the bin evenly so that the simulation results between two simulations can be meaningfully compared. We do not use crossover operation for input data calibration as it would result in the number of vehicles in the simulation to be not conserved.

We only show the results for calibration of the input data in this paper. The model parameter values are retained as in the original NaSch-model (Nagel and Schreckenberg, 1992) as $v_{\max} = 5$, $a = 1$, $d = 1$ and $p = 0.3$. These parameters were not included as part of calibration procedure but fed to the simulation instead. Having set the model parameters we optimized the GA for finding the right set of input data Φ_A such that the traffic count at position C will match the generated ground-truth data. We use root mean squared error (RMSE) as cost function for the optimization procedure. Since there is inherent randomness in the NaSch-model, we

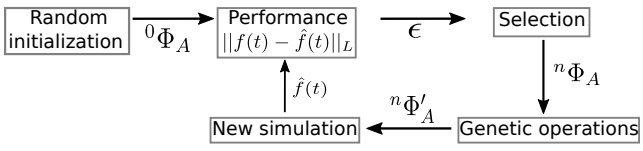

Figure 3: Procedural steps in genetic algorithm

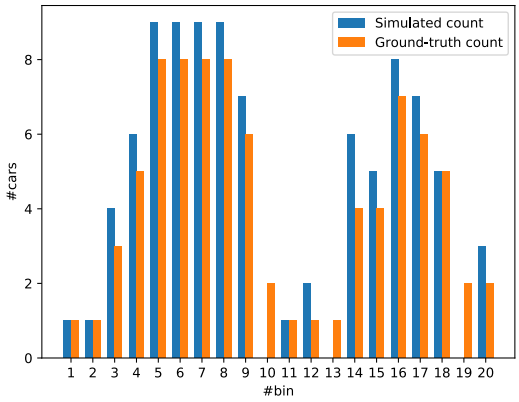

Figure 4: Single lane traffic flow distribution generated at point C compared to synthetically generated ground-truth.

ran 20 simulations for each input data and selected the solution that generated distribution at point C closest to the average value of the 20 distributions.

We selected population size of 32 for GA with 4 *elite* solutions. Elite solutions are the best performing solutions in the population which as retained in the next iteration of GA to prevent loss of good solutions in the search. Fig. [4] shows the calibration result of the best solution of GA after 3000 iterations. The best individual has the RMSE error of 1.12 indicating that there is on average 1 mismatched vehicle per bin. We claim that this accuracy in the solution is reasonable. Due to the stochastic nature of NaSch-model, it is not possible to resolve the error below 1 mismatched vehicle per bin. The exception to this rule are the deterministic limits of $p = 0$ and $p = 1$ in the NaSch-model.

We note that due to the stochastic nature of probabilistic CA, the calibration of input data becomes increasingly more unreliable as we increase the simulation time and space (distance between point A and C). Furthermore, the variation in distribution generated for different runs with same input data Φ_A increases as we increase the value of state transition probability in the model.

In future work, we will demonstrate a complete calibration procedure that will fine tune not only the input data but also the model parameters simultaneously. We will also investigate how well can we scale up the P-CA models such that calibration and validation can be performed on the real world highway sections.

References

Asteroth, A. and Hagg, A. (2015). How to successfully apply genetic algorithms in practice: Representation and parametrization. In *2015 International Symposium on Innovations in Intelligent SysTems and Applications (INISTA)*, pages 1–6. IEEE.

Nagel, K. and Schreckenberg, M. (1992). A cellular automaton model for freeway traffic. *Journal de physique I*, 2(12):2221–2229.