J Supercomput (2013) 65:664–681
DOI 10.1007/s11227-013-0952-5

Cellular automata on FPGA for real-time urban traffic signals control

G. Kalogeropoulos · G.C. Sirakoulis · I. Karafyllidis

Published online: 4 May 2013

Abstract Among different traffic features, the urban traffic has received a lot of attention due to the ongoing traffic congestion as a result of increased car usage, population growth, and changes in population density. In urban networks, the vehicles flow differs when compared with highways flow because of the freeway's low speed limit but mostly because of the traffic lights control. In this paper, a real-time hardware implemented bio-inspired model for traffic lights control is presented. The proposed model arrives from Cellular Automata (CAs), which have been proven very flexible and powerful computational traffic models, in that they are able to capture all previously mentioned basic phenomena that occur in traffic flows. The resulting CA model was hardware implemented on FPGA to take full advantage of the inherent parallelism of the CAs and to support the function of an advanced electronic system able to provide real-time adaptive control of traffic lights designed to consider traffic conditions for the whole intersections. The analytical results, obtained by application of the aforementioned FPGA CA processor are found in excellent agreement with the numerical simulations.

Keywords Traffic signals · Cellular automata · FPGA · Real-time control

1 Introduction

For the past century, cars have been playing an important role in changing the city life by accelerating the outward expansion of population into the suburbs, but also

G. Kalogeropoulos · G.C. Sirakoulis (✉) · I. Karafyllidis
Department of Electrical and Computer Engineering, Democritus University of Thrace, Xanthi 67100, Greece
e-mail: gsirak@ee.duth.gr

G. Kalogeropoulos
e-mail: georkalo@ee.duth.gr

I. Karafyllidis
e-mail: ykar@ee.duth.gr

by introducing numerous traffic issues. The suburban trend is emphasized by the fact that highway transportation encourages business and industry to move outward to sites where access by car is easier. On the other hand, the rise of car usage and the corresponding growth of suburbs that have allowed people to live on the outskirts of the city and move to the crowded cities on a daily basis, gave also rise to problems, such as the traffic jam, traffic bottlenecks, traffic accidents, air pollution, etc. that became prominent especially in metropolitan centers. In specific, resulting frequent road traffic congestion has been increasing worldwide during the last decades as a result of increased car usage, population growth, and changes in population density [17]. Consequently, one of the growing topics for intense research activity is traffic modeling, which includes among others vehicular traffic on highway networks, city traffic with signal control, and public conveyance. The investigation of such traffic systems is important for both practical and scientific reasons. Reducing traffic congestion contributes to economic efficiency and modeling and analysis of traffic flow allows for new insights into development of effective and efficient intelligent transportation management system.

Among different traffic features, the urban freeway traffic has received a lot of attention that is attributed to it being a particular kind of freeway [15]. In specific, urban freeway shows different phenomena compared with those on the highways because of the freeway's low speed limit and mostly because of the fact that vehicles flow in urban networks is almost entirely controlled by traffic lights [13]. Generally, traffic lights controls are implemented with pretimed or actuated or adaptive control. A pretimed controller repeats preset signal timings derived from historical traffic patterns, but an actuated controller computes phase durations based on real-time traffic demand obtained from the detection of passing and stopped traffic on all lanes leading into an intersection [31]. Finally, adaptive control is designed to consider traffic conditions for the whole intersection and is capable to adjust signal phasing and timing settings in response to real-time traffic demands at all or some approaches [23]. Testing control strategies in reality is usually infeasible or at least extremely demanding in time and costs; therefore, the application of simulation tools as laboratory environment is desired [9]. Respectively, the development of efficient traffic signals control models may be very beneficial when studying various kinds of city networks, even those with a more sophisticated topology [19].

Classically, as in case of modeling of any real physical system, the traffic modeling starts with the selection of the observation and representation scales. In the field of traffic flow modeling, there are two different concepts for vehicular traffic simulation, i.e., macroscopic or "coarse-grained" point of view and microscopic description. In the former, due to analogies of traffic flow with gas-kinetic theory (Boltzmann-like, [30]) and fluid dynamics [21], the traffic is viewed as a compressible fluid formed by vehicles and the resulting macroscopic traffic models based on lax analogies with the equations of ordinary fluids or are of a more or less phenomenological nature [6]. Therefore, it is not surprising that, although all these models are considered to be suitable for describing traffic phenomena on freeways, each of them shows serious deficiencies in others [16].

On the other hand, in microscopic approximation, traffic is treated as a system of interacting particles where attention is explicitly focused on individual vehicles

and the interactions among them. Consequently, these models were considered much better suited for the investigation of especially urban traffic [4]. Microscopic models include follow-the-leader car models in which it is assumed that the acceleration is determined by vehicles in front of the driver, and Cellular Automata (CAs) programming paradigm from statistical physics in which each vehicle is represented by an occupied cell in a CA model. The corresponding CAs traffic models are very flexible and powerful, in that they are also able to capture all previously mentioned basic phenomena that occur in traffic flows [5]. In a larger setting, these models describe self-driven, many-particle systems, operating far from equilibrium and in contrast to strictly gaseous analogies, the particles in these systems are "intelligent" and able to learn from past experience, thereby opening the door to the incorporation of behavioral and psychological aspects [49]. As a result, in 1992, Nagel and Schreckenberg proposed a CA model, the well-known NaSch model of road traffic, that was able to reproduce several characteristics of real-life traffic flows [26]. Although the NaSch model reproduces the basic structure of the fundamental diagram and the appearance of spontaneous jam formation, it does not exhibit metastable states of high flow and synchronized traffic. As a result, the VDR model and the CA model of synchronized traffic were introduced [33]. Despite the huge number of CA models, several attempts have been made to review and categorize the proposed CA models from different points of view [27, 32, 33]. In [24], the CAs traffic models are categorized as single and multiple cells models. In specific, the CAs single cell models are distinguished as deterministic, like Wolfram's rule 184 (CA-184) and the Fukui-Ishibashi CA model, stochastic models, like NaSch and mean-field theory models [34] and slow-to-start models such as Takayasu and Takayasu model [43] and the Benjamin, Johnson, and Hui (BJH) model [2], as well as the Nishinari et al. model [29]. On the other hand, the multicell models still refer to single-lane traffic, where a vehicle is allowed to span a number of consecutive cells in the longitudinal direction, have encountered among others the Helbing–Schreckenberg model, Brake-light CA model as well as the model of Kerner, Klenov, and Wolf [24].

At about the same time when the NaSch CA model was introduced [26], another CA model focused on city traffic was suggested by Biham, Middleton, and Levine (BML) [3]. This two-dimensional (2-d) model is rather simplistic since it only consists of the intersections without any connecting streets. It should be mentioned that because of the effect of traffic lights, urban roads as well as the corresponding CA models have special characteristics [42]. Consequently, Guoging et al. extended the BML model by revisiting the regulation on traffic lights [14] and rearranging the BML rules [10]. The study and improvement of network efficiency has aroused until today much concern [46], while modern optimization techniques have been used for the parameterization of CA rules [24].

Among several CA-based mobility models for urban traffic, the model of Wei et al. [47] is the most similar to our work. The most intriguing part of the aforementioned model is the fact that used the original characteristics of the CA microscopic model to successfully describe and handle macroscopic properties of traffic streams resulting in an almost macroscopic CA model. However, the proposed here model provides some new characteristics to cope better with some of the previous model arguable features, for example, the original implementation uses a Poisson distribution for the incoming boundary directions flow pressure values, which although

sounds reasonable does not provide realistic situations of vehicular traffic flows near the studied intersections. On the other hand, the presented CA model is characterized by as much as low complexity as possible so that the computational recourses are kept low while its computation speed is kept high without losing any of the requested essence of complexity regarding the real-time signals control of urban networks intersections. Moreover, with regards to realism, the use of a linear feedback shift register (LFSR) driven by a 1-dimensional CA able to produce pseudo-random values greatly enhances the vehicular traffic flow simulation in order to produce as much life-like traffic conditions as possible. As a result, the proposed here model focused on traffic light control strategies and tried to find optimal model parameters in order to maximize the network flow. Furthermore, because of the inherent parallelism of CAs, one of the most pronounced features of the introduced model is its straightforward hardware implementation with the help of Very High Speed Integrated Circuit (VHSIC) Hardware Description Language (VHDL) synthesizable code resulting to speed up the CAs application to the real-time traffic signals control. It should be mentioned that CAs are one of the computational structures best suited for hardware realization. In this paper, the design processing of the finally produced VHDL code, i.e., analysis, elaboration, and simulation, has been checked out with the help of the Quartus II, v. 9.0 design software of the ALTERA Corporation and the use of the high-end Altera Stratix and Stratix III series chips. The proposed hardware was optimized in the areas of logic elements utilization, maximum frequency, and chip area coverage and was measured successfully against modern CPUs. Consequently, the proposed here FPGAs outperform the corresponding software implementations of the CA model. The resulting hardware can be considered as basic component of an advanced electronic system able to support the function of an advanced electronic system responsible to provide real-time adaptive control of urban traffic lights designed to consider traffic conditions for the whole intersections. The analytical results, obtained by iterative application of the aforementioned FPGA CA processor are found in excellent agreement with the numerical simulations. As a result, the proposed FPGA could serve as the basis of a support decision system for monitoring urban traffic lights function in real-time, providing valuable near optimum control services.

2 Cellular automata preliminaries

Cellular Automata (CAs) [28] are models of physical systems, where space and time are discrete and interactions are local. They have been extensively used as models for complex systems and have also been applied to several physical problems, where local interactions are involved [1, 11, 24, 36, 37].

A CA consists of a regular uniform n-dimensional lattice (or array), usually of infinite extent. At each site of the lattice (cell), a physical quantity takes on values. The value of this physical quantity over all the cells is the global state of the CA, whereas the value of this quantity at each site is its local state. Each cell is restricted to local neighborhood interaction only and, as a result, it is incapable of immediate global communication [28]. The neighborhood of a cell is taken to be the cell itself and some (or all) of the immediately adjacent cells. The states at each cell are updated

simultaneously at discrete time steps, based on the states in their neighborhood at the preceding time step. The algorithm used to compute the next cell state is referred to as the CA local rule. Usually, the same local rule applies to all cells of the CA.

A CA is characterized by five properties:

1. the number of spatial dimensions (n);
2. the width of each side of the array (w). w_j is the width of the jth side of the array, where $j = 1, 2, 3, \ldots, n$;
3. the width of the neighborhood of the cell (r);
4. the states of the CA cells;
5. the CA rule, which is an arbitrary function F.

The state of a cell, at time step $(t + 1)$, is computed according to F, a function of the state of this cell at time step (t) and the states of the cells in its neighborhood at time step (t). For a 2-d CA, two neighborhoods are often considered: Von Neumann, which consists of a central cell and its four geographical neighbors north, west, south, and east. The Moore neighborhood contains, in addition, second nearest neighbors northeast, northwest, southeast, and southwest, which is a total of nine cells. In most practical applications, when simulating a CA rule, it is impossible to deal with an infinite lattice. The system must be finite and have boundaries. Clearly, a site belonging to the lattice boundary does not have the same neighborhood as other internal sites [7]. In order to define the behavior of these sites, neighborhood is extending for the sites at the boundary, thus leading to various types of boundary conditions such as periodic (or cyclic), fixed, adiabatic, or reflection.

CAs have sufficient expressive dynamics to represent phenomena of arbitrary complexity and, at the same time, can be simulated exactly by digital computers because of their intrinsic discreteness, i.e., the topology of the simulated object is reproduced in the simulating device [40]. The CA approach is consistent with the modern notion of unified space-time. In computer science, space corresponds to memory and time to processing unit. In CAs, memory (CA cell state) and processing unit (CA local rule) are inseparably related to a CA cell [35, 38]. Furthermore, they can easily handle complicated boundary and initial conditions, inhomogeneities, and anisotropies [8]. In addition, algorithms based on CAs run quickly on digital computers [44]. Models based on CAs lead to algorithms, which are fast when implemented also on serial computers because they exploit the inherent parallelism of the CA structure. These algorithms are also appropriate for implementation on massively parallel computers [41], such as the cellular automaton machine (CAM) [48] or FPGAs [12, 18, 25].

3 CA model for urban traffic signals control

As mentioned before, in many places, the capacity of the road traffic system is regularly exceeded by the traffic demand. As a possible solution to the problem the physical expansion of the road infrastructure has been proposed. However, such a solution presupposes long-term building programs, it is very expensive, can only be accomplished in the long term and could be very environmentally unfriendly. Another solution could be the reduction of the traffic demand, for example, by stimulating

alternative transportation modes, pricing and parking regulations; however, it also results in more expensive traveling, and thus is strongly disapproved of by users of the traffic system. An also interesting approach to tackle with the problem is to attempt improved on-line matching between infrastructure supply and traffic demand. Given the dynamics of the traffic system, the traffic operators should and could know what the current traffic situation is and how it will evolve in the near future. Since the different kind of traffic measurements only provide incomplete information, there is an element of prediction in both the forecast and the "now-casting." Taking into consideration the tracking and tracing of individual vehicles and the information collected by floating cars as well as the traffic signals control, a CA traffic model is proposed in the foregoing, aiming to enrich the available information on the actual traffic situation and provide traffic prediction and optimal traffic management under feed-back and control.

In the proposed CA traffic model, each traffic intersection is regarded as a cell in an urban traffic signal network. Each intersection is modeled as a cell with a Von-Neumann neighborhood domain. Each approach is appointed a number from 1 to 4 in a clockwise order, with 1 being the west approach of the intersection. All intersections can then be put in a 2-d array with size $N \times M$, where N is the number of rows with each row representing an intersection and M the number of columns where each column represents the corresponding approach and their value of the resulting flow pressure (FP), respectively.

Regarding the time evolution of the CA cell to its next state it must first gather information about the state of its neighbor cells; in this case, the flow pressure of each direction. This phase is called the state perception phase. At this point, it should be made clear that in the considered network, all streets are equal in respect to the processes at intersection, in other words no streets or directions are dominant. Furthermore, it should be also noticed that the free flow phase of the proposed method is an artifact of the periodic boundary conditions and of the fact that no vehicle turns. In more realistic situations, if an intersection is blocked by, e.g., an accident, the method would not allow the blockage to spread to other intersections by blocking flow into the affected intersection. Finally, pedestrians are not considered in the presented model and the critical time for safe crossing of the under study intersection, at least a period of 20 sec, should be accordingly implemented in the model manually by the user by fine tuning the corresponding parameter. As a result, traffic light periods for all streets (intersections) are assumed to be equal in the following.

Assuming sensors measure the flow pressure taking continuous values that range from 0 to 1, according to the following expression:

$$\mathrm{FP} = \frac{Z \times S}{L \times K} \tag{1}$$

where Z is the cars number, S the average length of the car, L the measurable road length, and K the number of lanes of each approach. The flow pressure readings are then inserted into the intersection array so that each cell has access to the states of its neighbors. As a result, the state of an intersection would be finally given by the following equation:

$$P_i^{t+1} = R\left(P_i^t, P_{i1}^t, P_{i2}^t, P_{i3}^t, P_{i4}^t\right) \tag{2}$$

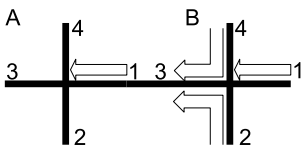


Fig. 1 Intersection flow pressure

where t stands for the time step evolution and R denotes the state CA rule. In general, the same rule with corresponding directions can be applied to every intersection. More specifically, the outline of the proposed rule, for the under study intersection, can be summarized to the following:

1. *If* the normalized sum of the three neighbors, i.e., potential directions [taking into account for each of them, their impact to the final decision (through, left and right, respectively)] is greater than the sum of the specific direction flow with the minimum flow increment, i.e., 0.1;
2. *Then* this specific direction flow will be raised by the aforementioned minimum flow increment, namely 0.1;
3. *Else* it will not change during this time step.

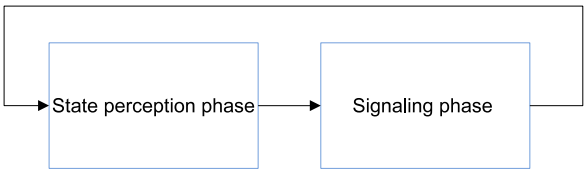
In the above rule description, let us name W_k the total impact normalization coefficient to the local intersection after k normalizations that have passed according to the empirical function:

$$W_k = (10 - k) \times \left(\frac{0.5}{10}\right) + 0.5 \tag{3}$$

where W_k takes values from range [0, 1] and as the generations progress, it is gradually reduced in steps of 0.05. This is to simulate the decrease of impact that the flow pressure of faraway neighboring intersections will result on the current intersection after k normalizations. The constants found in Eq. (3) resulted optimized after several tests.

Consequently, the state perception phase results then from different rounds of normalization in order to proceed to the next time step and evolve each CA cell state. At this point, it should be also mentioned that each of the three possible neighbor directions contributes by the corresponding coefficients, respectively. For example, as shown in Fig. 1, in order the intersection A cell to update its state all eastern, starting clockwise, flow pressure from intersection B must be taken into account, and in this case, w_0, w_1, and w_2 would be the resulting impact coefficients of each direction (straight, left and right turn, respectively) with their sum always equal to 1. In order to evaluate the proposed model, Matlab code was developed to simulate the time step evolution of the CA cell. In this, FP values were provided to the cell from all intersection directions with their respective impact coefficients and the evolution of the cell was monitored for all ten generations so that it would be in accordance with the above rule. Some software simulation results in conjunction with the imminent hardware implementation can be found in Fig. 9.

Fig. 2 Flow chart diagram



4 CA hardware implementation and simulation results

As mentioned before, in terms of circuit design and layout, ease of mask generation, silicon-area utilization, and maximization of clock speed, CAs are perhaps one of the most suitable computational structures for VLSI realization [39]. More specifically, from circuit designing point of view, there are four main factors that determine the cost/performance ratio of an integrated circuit, namely, circuit design and layout, ease of mask generation, silicon-area utilization, and maximization achievable clock speed; for a given technology, the latter is inversely proportional to the maximum length of the signal paths. CA circuit design reduces to the design of a single, relatively simple cell and layout is uniform. The whole mask for a large CA array (the cells with their internal connections as well as the interconnection between cells) can be generated by a repetitive procedure so no circuit area is wasted on long interconnection lines and because of the locality of processing, the length of critical paths is minimal and independent of the number of cells.

The proposed algorithm consists of two stages as shown in the diagram (Fig. 2). The first one is the *state perception phase* which was described earlier. After this phase is complete, the *signaling phase* begins. Certain rules are applied for the traffic signal control. In order of descending priority these are:

- Change the signal to green if red is shown to this direction for `RED_M` consecutive times.
- Change the signal to green to the direction in which the previous signal was red and has the highest flow pressure.
  - The flow pressure must be at least `Red_t`.
- If the signal is green for the `Green_Max` consecutive time in the same direction then change the signal to red.
- If none of the above applies then change the signal to green to the approach with the highest flow pressure.

All variables used (`RED_M`, `Red_t`, `Green_Max`) are preset. In order to increase the functionality of the presented CA model, two more stages to the original diagram were added. The first one refers to the *CA model initialization* in which values are inserted in the intersection data so that the simulation does not begin with no traffic at all in every intersection. The second one is the usage of *linear feedback shift register (LFSR),* which is responsible for the inbound traffic simulation coming through the boundary directions, based on pseudorandom values in order to represent the flow pressure of the boundary directions. Regarding the generation of these pseudorandom variables, a *one-dimensional* (1-*d*) *CA for pseudorandom number generation* [22] is used. This generator is based on the real time clock sequence (analytical time

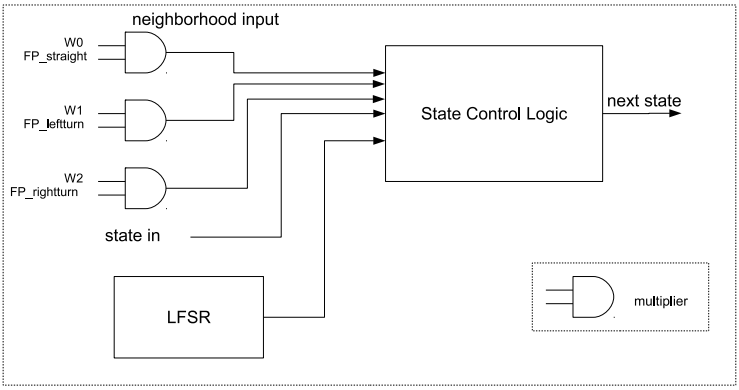


Fig. 3 Block diagram of the CA cell architecture

description) and can generate high-quality random numbers, which can pass all of the statistical tests of DIEHARD as well as NIST. More details regarding the usage of the aforementioned CA Pseudorandom Number Generator (PRNG) can be found on [22].

In general, some of improvements found in the proposed model can be summarized as follows. The signaling computations were done according to intervals, where each interval equals the time needed for calculating the next CA cells states and the appropriate signaling values without constraining them in time. If needed, a maximum time constraint can be easily added. Furthermore, the original implementation uses a Poisson distribution for the incoming boundary directions flow pressure values whereas in the FPGA a LFSR is used with pseudorandom values to simulate the incoming traffic and in order to provide realistic situations of vehicular traffic flows near the studied intersections. More over, simulating traffic over a period of time with random values of a distribution emulating incoming boundary traffic does not guarantee that every possible traffic scenario will be tested. So, in order to check every possible signaling rule, signals were carefully selected so that every rule and possible traffic situation can be tested and handled appropriately. Initially, the width of the road and car volume was passed to the software to calculate the flow pressure values. However, in the presented FPGA implementation, the used sensors automatically calculate the flow pressure (FP) values and reduce it accordingly. Finally, the flow pressure takes continuous values that range from 0 to 1, according to Eq. (1).

The implementation of the CA model was developed with VHDL using the Altera Quartus II software. The schematic design of the corresponding CA cell is depictured in Fig. 3. The parameters used in the development and simulations of the code were in accordance with the ones found in [19]. More specifically, the initial number of intersections was set to 6 and arranged in 2×3. The initial flow pressure values in the intersection array were chosen randomly on a scale from 0 to 100 with the help of the aforementioned 1-d CA PRNG. The boundary approaches flow pressures are attributed through a 4-bit linear feedback shift register (LFSR), while the values of signals `RED_M`, `Red_t` and `Green_Max` are set to 8, 60, and 2, respectively. Furthermore, highest priority is given to the east priority with descending priority

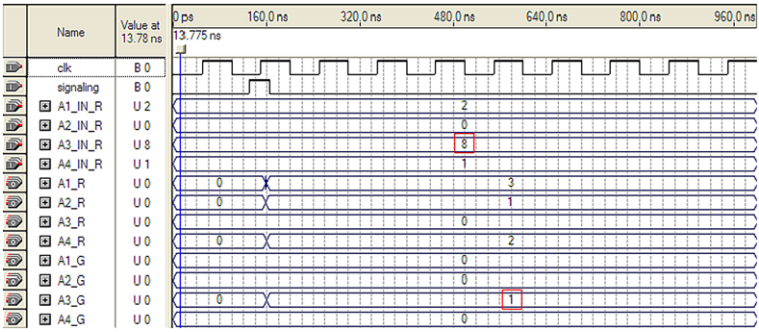


Fig. 4 Application of consecutive red signaling rule. The accumulation of 8 consecutive red signals trigger the green signal for approach 3 (east)

given to the remaining directions in a clockwise order. However, the selected priority is user defined and can be easily updated with the proper signal. As a result, if the rule for `Red_t` flow pressure applies then the order is counterclockwise. Finally, the cell state evolving generations is set to 10 and the values selected for impact coefficients w_0 is 0.8 and for w_1 and w_2, respectively.

The state perception phase was also simulated with Quartus II for many different scenarios. It is noted that through generations 5 to 10 no changes to the flow pressure are made due to the relatively small number of intersections which causes the vectors to stabilize to their values earlier. After all the normalizations of the state perception phase are complete, a signal is raised in order for the traffic signaling phase to begin. The simulation results of the signaling rules in intersection A are described below. The `Ax_IN` inputs represent the intersection's directions flow pressures, the `Ax_IN_R` and `Ax_IN_G` equal to the number of consecutive red and green signals, respectively, and the `Ax_R` and `Ax_G` the red and green signals on the intersection. One can observe that input `A3_IN_R` equals 8 which is the threshold for the maximum number of consecutive red signals. As expected, the green signal is shown to the western approach [Fig. 4, signal `A3_G`].

In the next case, the western approach's flow pressure is 60 which is greater than the threshold set which triggers the `Red_t` signaling rule. The next rule in descending priority is the `Green_Max` rule when 2 consecutive green signals are given in the same direction. Then a red signal must be given to this direction and decide according to the maximum flow pressure of the other directions. It is shown in Fig. 5 that the red signal is given to the southern approach (number 2) because of the `Green_Max` rule and the green signal is decided by the flow pressure values of the rest approaches. Thus, the green signal is given to the eastern approach with a FP value equal to 40. Finally, when none of the above rules is valid, the green signal is decided by the maximum flow pressure.

The exact FPGA device used for the implementation is Altera Stratix, device EP1S60F1020C5. This FPGA uses 130 nm technology and it allows the use of up to 57,120 logic elements [19]. In addition to the VHDL implementation, Matlab code was also developed in order to compare with the FPGA performance. The CPU used for the simulations is an AMD Phenom II 920 Quad-Core CPU clocked at 2.8 GHz.

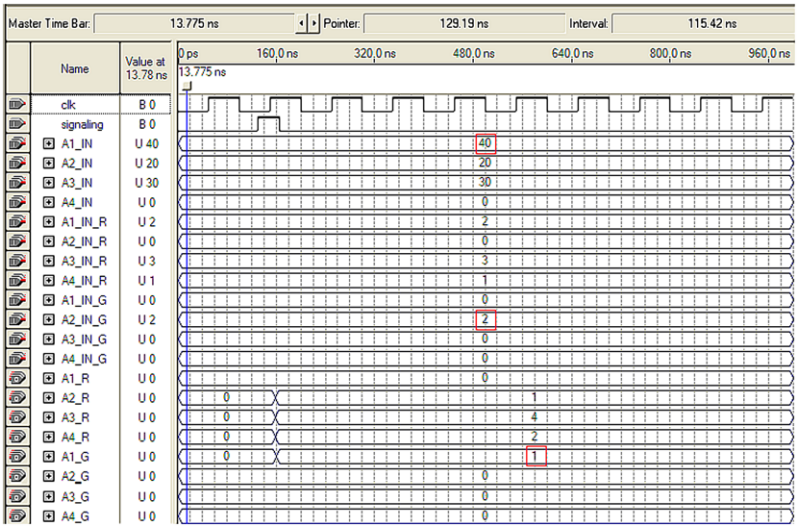


Fig. 5 Two consecutive green signals for the same approach (2) triggering the `Green_Max` signaling rule

Software based simulations are limited in sequential machines, since the inherent parallelism of the CA has to be emulated. Typically, this is achieved by calculating the time evolution of each cell separately and using double buffers to simulate the parallel nature of CA, thus leading to a considerable slowing down of the simulations [47]. The CA cell rules in this simulation are preset so the average number of instructions which have to be executed can be calculated. An average of 2,550 instructions (adding, comparing, subtracting, multiplying, etc.) were required for calculating the next cell state and signaling. This means that even with a super scalar CPU architecture the necessary clock cycles are 159 whereas the FPGA only requires 10 cycles for the output signals to be calculated. In the case of Matlab simulation, the mean completion time was 280 ns while 240 ns were required for the FPGA to complete the necessary calculations. The FPGA's native parallelism in executing commands in conjunction with the inherent parallelism of the CAs allows us to presume that in larger systems, which include a vast number of intersections the FPGA's ability for parallel processing can give a significant advantage over a general purpose CPU, which makes it suitable for this controller.

New cell design and differences with the previous one In order to enhance the hardware performance of the proposed FPGA designs, various changes were made to the initial FPGA design, which finally resulted in greater speed and efficiency. In specific, the CA cell now directly communicates only with its 4 neighboring cells (Fig. 6). That way the design is much more flexible and efficient due to the fact that every possible arrangement of intersections can be accommodated and is not confined by the original 2×3 intersection layout design. Moreover, the single CA cell takes up much less space and gates on the chip itself which translates into shorter electronic routes, higher speeds and less energy consumption. The resulting advantages can be

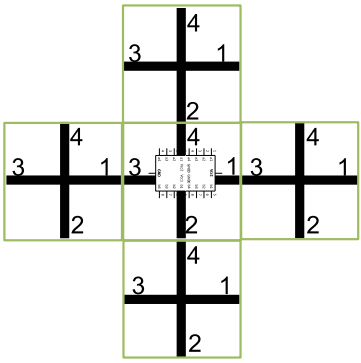


Fig. 6 Updated CA cell architecture with four neighbor intersection layout

Table 1 FPGA devices comparison of resources usage

	FPGA Devices		
	Old design Stratix	New design Stratix	New design Stratix III
FPGA Chip	EP1S60F1020C5	EP1S60F1020C5	EP3SL50F484C2
Max Frequency	42 MHz	63.31 MHz	144.36 MHz
Total logic elements	52,535/57,120 (92 %)	466/57,120 (<1 %)	466/95,000 (<1 %)
Total pins	214/728 (27 %)	145/728 (19 %)	145/296 (49 %)
Total virtual pins	0	0	0
Total memory bits	0/5,215,104 (0 %)	0/5,215,104 (0 %)	0/1,880,064 (0 %)
DSP block 9-bit elements	0/144 (0 %)	16/144 (9 %)	16/216 (7 %)
Total PLLs	0/12 (0 %)	0/12 (0 %)	0/4 (0 %)
Total DLLs	0/2 (0 %)	0/2 (0 %)	0/4 (0 %)

clearly seen centralized in Table 1, which compares the previous and new design. Another advantage of the new design is that more complex intersections can be handled in comparison with other proposed methods FPGA implemented such as the BML method, which incorporates only some of the possible directions of vehicle movement in intersections [20]. Furthermore, nontraffic related traffic light points, such as pedestrian crossings, can be also easily managed. Many times these crossings coexist with traffic lights at intersections and now they can be easily handled in the same way as the vehicle traffic flow with appropriate signaling.

The new design was tested in two different FPGA chip models due to the fact that it required a lot less space and utilized less logic elements, which made it suitable for a wider range of chips. The first test was conducted on the same chip model as the original design so that the differences of the new layout could become apparent. The most obvious change is the logic elements utilization which is 100 times less than the original. This means that the new cell can be accommodated in smaller chips and this way the critical paths become shorter with an increase in speed, and at the same time less heat dissipation and energy consumption. Fewer pins are also required for the signals and that way smaller, more efficient, and less expensive chips can be used. The increase in speed when the same as before Stratix chip is used can be up to

Table 2 Software—hardware comparison

	Old design Stratix	New design Stratix	Improvement (old vs. new design)
Software (Matlab)	280 ns	81.91 ns	341 %
Stratix FPGA	240 ns	15.87 ns	1512 %
Stratix III FPGA	–	6.98 ns	3438 % (compared to Stratix)

150 % and when the new Stratix III chip is utilized it skyrockets to 343 %. The speed increase, which derives from the inherent parallelism of the FPGA architecture, can be more obvious if it is compared to a modern CPU. That is why the same model was implemented on Matlab and tested on an Intel Core i7-2670QM quad-core processor running at 2.2 GHz.

Comparing the two proposed designs, it is shown that when the previous design was used [19], the times for the software and the hardware to complete the process were almost comparable and so the benefits from using an FPGA were not so obvious. The proposed here design has greatly enhanced the chip's overall efficiency and the completion times of both the software and hardware implementation have dropped significantly as shown in Table 2. Especially, the improvement in the Stratix chip is an average 1512 % and if the faster Stratix III is used then the impact is even greater and can reach 3438 % compared to the previous design running on a Stratix chip. It should be made clear the enhancement of FPGA performance results mostly from the proposed design improvements and not from the possible usage of new FPGA devices. Furthermore, the FPGA completes the perception phase at 15.87 ns compared to 81.91 ns of the software edition which is more than 5 times faster. When compared with other FPGA implementations of CA traffic models that incorporate parallelism in their designs, which achieved a 340 % speed up to the software results, the proposed here design can achieve a far greater acceleration of almost 4000 % the CPU-processed simulation [45]. Apart from the state perception phase the algorithm which is implemented in the FPGA also consists of the signaling phase, which originally, had to be programmed into a separate chip because of the number of the gates and the overall chip space the state perception phase required [19]. With the proposed here, design both phases can be accommodated in the same chip with obvious advantages in speed, efficiency, cost, energy consumption and complexity of the system. Another useful aspect of the new design is the very low utilization of the chip resources. Since only 1 % of the logic elements is being used and there is no need for an array to store the FP values then it is very easy and straightforward to implement changes to the code so that more intersections can be accommodated in the same chip.

In order to check the integrity of the new design approach, simulations with random values were conducted using Quartus II. In these, random flow pressure values were fed to the FPGA in all four directions so that they could trigger a change in the FP values and so to check that the evolution rule was implemented correctly without any conflicts (Fig. 7). As mentioned before, this task was accomplished with the help of the proposed 1-d CA PRNG [22]. The range of the values which were used consisted of integers from 0 to 100 representing the inbound traffic of each direction in a clockwise manner (North, East, South West) or anticlockwise depending signaling.

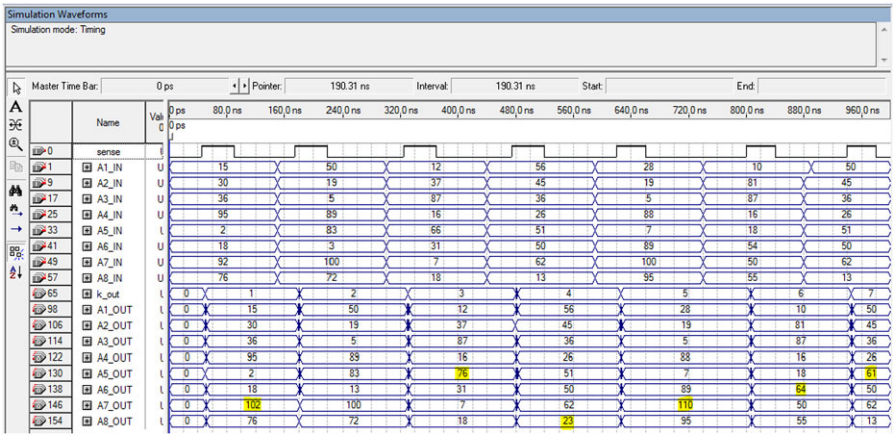


Fig. 7 Random Values Simulation

At first, the rule was crosschecked only for 1 out of the 10 generations required. This was done on one hand because most changes occur on the first generation of the CA model and on the other hand in order to confirm that the rule was followed correctly because after 3 or 4 generations the vectors for each intersection were stabilized and the changes would no longer be visible. The highlighted values indicate that the inbound traffic flow to this direction of the intersection was high and the corresponding FP value according to the time step evolution rule has increased. This event is then evaluated in the next stage of the algorithm, which is called the signaling phase and could trigger a green light for this direction according to the signaling rules set. In the next set of simulations the same set of random values for each direction was evolved for all 10 generations of the rule. The eastern approach due to the very low flow in the beginning is expected to rise in value because of the incoming flow from all the other directions and especially the south which is 92, close to the maximum of 100. This is confirmed in the simulation as the eastern approach (`A5_OUT`) flow pressure rises from 2 to 32 in the first 3 generations of the CA model evolution. The changed values of the eastern approach are being highlighted in red inside Fig. 8. It is also evident that after the 4th generation the FP value vector stabilizes and no further changes are made until the 10th generation when the state perception phase is stopped and the flow pressure values are then used in the signaling phase.

The same scenarios that were simulated with the help of the aforementioned Matlab code in Sect. 3, were also implemented on the FPGA chip. That way, the hardware implementation could be crosschecked of its integrity and the validity of its results. The outputs of both the software and hardware implementations were found to be in excellent agreement with each other for all generations as depicted in Fig. 9. For clarity of the aforementioned results, it should be mentioned that in VHDL, specialized libraries are required to handle decimal numbers and, therefore, increasing complexity and silicon area utilization. To tackle with the aforementioned complexity in the hardware implementation, FP values, which are in the range of [0, 1], were accordingly normalized to integrals in the range of [0, 100].

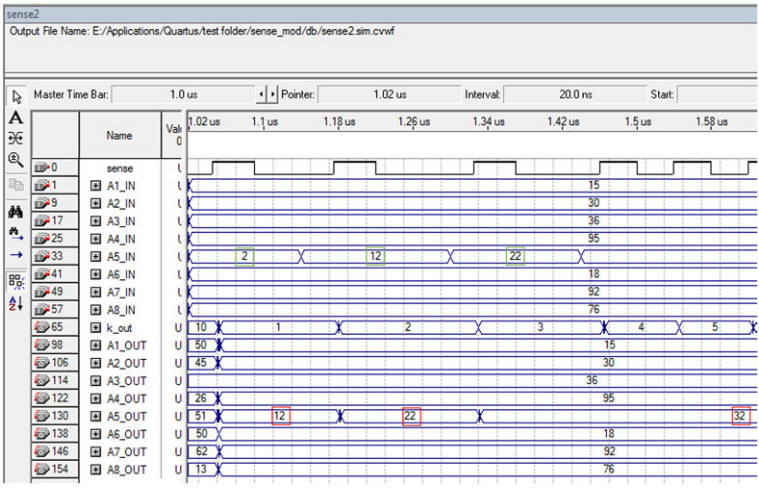


Fig. 8 10th generation simulation results

Finally, it should be noticed that the proposed design can be easily adopted for various traffic applications and all components can be changed to suit various urban network traffic conditions. All variables used can be automatically parameterized through the architecture section so they can be easily changed to the desired-requested values. Furthermore, CA traffic several controllers, as the one proposed here, can be connected, each one controlling numerous intersections and feed each other with traffic information to form a larger, more complex, and more versatile traffic signaling network. In specific, the proposed hierarchical CA structure adopts the following scheme where every overlying CA cell (at the top of the hierarchy), which corresponds to several connected crossroads is described as a new partial CA grid consisted of these several underlying CA cells (lower at hierarchy) that correspond to the aforementioned crossroads.

5 Conclusions and future work

In this paper, a CA model able to successfully describe and handle macroscopic properties of traffic streams was implemented in FPGA to take advantage of the inherent parallelism of CAs. The presented here model focused on traffic light control strategies and tried to find optimal model parameters in order to maximize the network flow. The resulting hardware was optimized in the areas of logic elements utilization, maximum frequency and chip area coverage and was measured successfully against modern CPUs. Consequently, the proposed here FPGAs outperform the corresponding software implementations of the CA model. The analytical results, obtained by iterative application of the aforementioned FPGA CA processor are found in excellent agreement with the numerical simulations. As a result, the proposed FPGA could serve as the basis of a advanced electronic support decision system for monitoring traffic lights function in real-time, providing near optimum control services and considering traffic conditions for the whole intersections. As future work concerns, it

Simulation Results

Generation	1 (Matlab)				1 (Quartus)				3 (Matlab)				3 (Quartus)				10 (Matlab)				10 (Quartus)			
Intersection / FP Values	East	South	West	North	East	South	West	North	East	South	West	North	East	South	West	North	East	South	West	North	East	South	West	North
A	40	0	20	30	0.4	0	0.2	0.3	50	30	20	30	0.5	0.3	0.2	0.3	50	30	20	30	0.5	0.3	0.2	0.3
B	60	40	10	80	0.6	0.4	0.1	0.8	60	40	10	80	0.6	0.4	0.1	0.8	60	40	10	80	0.6	0.4	0.1	0.8
C	10	30	90	20	0.1	0.3	0.9	0.2	10	50	90	20	0.1	0.5	0.9	0.2	10	50	90	20	0.1	0.5	0.9	0.2
D	30	80	70	10	0.3	0.8	0.7	0.1	30	80	70	20	0.3	0.8	0.7	0.2	30	80	70	20	0.3	0.8	0.7	0.2
E	0	20	40	40	0	0.2	0.4	0.4	30	20	50	60	0.3	0.2	0.5	0.6	40	20	50	60	0.4	0.2	0.5	0.6
F	50	60	20	50	0.5	0.6	0.2	0.5	50	60	40	50	0.5	0.6	0.4	0.5	50	60	40	50	0.5	0.6	0.4	0.5

	Matlab	**Quartus**
Number of transitions:	18	18
Generation of transitions	1, 2, 3, 4, 5	1, 2, 3, 4, 5
Total change of FP	180	180
Total simulation time (ns	34.8	4.76

Fig. 9 Comparison results between software application and hardware implementation of the proposed CA model for the same scenario. The highlighted values show the transitions between each generation

would be interesting to compare the predictions of the proposed here CA model with empirical urban traffic traces and real-world traffic flow data selected by traffic cameras as found in metropolitan centers of Greece. Moreover, this, in turn, can verify the validity of the model or provide valuable feedback on how to further refine it in accordance with the aforementioned future support decision traffic control system.

References

1. Avolio MV, Crisci GM, Gregorio SD, Rongo R, Spataro W, Trunfio GA (2006) Sciara [gamma]2: an improved cellular automata model for lava flows and applications to the 2002 Etnean crisis. Comput Geosci 32(7):876–889
2. Benjamin SC, Johnson NF, Hui PM (1996) Cellular automata models of traffic flow along a highway containing a junction. J Phys A, Math Gen 29(12):3119
3. Biham O, Middleton AA, Levine D (1992) Self-organization and a dynamical transition in traffic-flow models. Phys Rev A 46:R6124–R6127
4. Brockfeld E, Barlovic R, Schadschneider A, Schreckenberg M (2001) Optimizing traffic lights in a cellular automaton model for city traffic. Phys Rev E 64:056132
5. Chowdhury D, Santen L, Schadschneider A (2000) Statistical physics of vehicular traffic and some related systems. Phys Rep 329(4–6):199–329
6. Daganzo CF (1995) Requiem for second-order fluid approximations of traffic flow. Transp Res, Part B, Methodol 29(4):277–286
7. D'Alotto L (2012) Cellular automata using infinite computations. Appl Math Comput 218(16):8077–8082
8. D'Ambrosio D, Spataro W (2007) Parallel evolutionary modelling of geological processes. Parallel Comput 33:186–212
9. Esser J, Schreckenberg M (1997) Microscopic simulation of urban traffic based on cellular automata. Int J Mod Phys C 08(05):1025–1036
10. Feng S, Gu G, Dai S (1997) Effects of traffic lights on ca traffic model. Commun Nonlinear Sci Numer Simul 2(2):70–74
11. Georgoudas I, Sirakoulis GC, Scordilis E, Andreadis I (2007) A cellular automaton simulation tool for modelling seismicity in the region of Xanthi. Environ Model Softw 22(10):1455–1464
12. Georgoudas I, Kyriakos P, Sirakoulis GC, Andreadis I (2010) An fpga implemented cellular automaton crowd evacuation model inspired by the electrostatic-induced potential fields. Microprocess Microsyst 34(7–8):285–300
13. Guan W, He S, Ma J (2012) Review on traffic flow phenomena and theory. J Transp Syst Eng Inf Technol 12(3):90–97
14. Guoqing G, Poming H, Binghong W, Shiqiang D (1998) Two-dimensional cellular automaton traffic model with randomly switching traffic lights. Appl Math Mech 19:807–813
15. He S, Guan W (2007) Empirical investigations on traffic phase transitions at Beijing ring road. In: Intelligent transportation systems conference, ITSC 2007. IEEE Press, New York, pp 290–295
16. Helbing D (1995) Theoretical foundation of macroscopic traffic models. Phys A, Stat Mech Appl 219(3–4):375–390
17. Institute TT (2011) Texas Transporation Institute. 2011 urban mobility report. http://mobility.tamu.edu/ums/report/
18. Jendrsczok J, Ediger P, Hoffmann R (2009) A scalable configurable architecture for the massively parallel gca model. Int J Parallel Emerg Distrib Syst 24(7):275–291
19. Kalogeropoulos G, Sirakoulis GC, Karafyllidis I (2011) Fpga implementation of a bioinspired model for real-time traffic signals control. In: Proceedings of the 2011 international conference on scientific computing, CSC 2011. CSREA Press, Las Vegas, pp 290–295
20. Ke P, Li Y, Nie X (2012) Self-adaptive optimization for traffic flow model based on evolvable hardware. In: Lei J, Wang F, Deng H, Miao D (eds) Artificial intelligence and computational intelligence. Lecture notes in computer science, vol 7530. Springer, Berlin, pp 255–262
21. Kerner BS, Konhäuser P (1994) Structure and parameters of clusters in traffic flow. Phys Rev E 50:54–83

22. Kotoulas L, Tsarouchis D, Sirakoulis G, Andreadis I (2006) 1-d cellular automaton for pseudorandom number generation and its reconfigurable hardware implementation. In: Proceedings of the 2006 IEEE international symposium on circuits and systems, ISCAS 2006, pp 4627–4630
23. Li H, Zhang L, Prevedouros PD (2008) Signal control for oversaturated intersections using fuzzy logic. In: Transportation and development innovative best practices 2008. American Society of Civil Engineers, Reston, pp 179–184. Chap 30
24. Maerivoet S, Moor BD (2005) Cellular automata models of road traffic. Phys Rep 419(1):1–64
25. Murtaza S, Hoekstra A, Sloot P (2008) Floating point based cellular automata simulations using a dual fpga-enabled system. In: Second international workshop on high-performance reconfigurable computing technology and applications, HPRCTA 2008, pp 1–8
26. Nagel K Schreckenberg M (1992) A cellular automaton model for freeway traffic. J Phys I Fr 2(12):2221–2229
27. Nagel K, Wagner P, Woesler R (2003) Still flowing: approaches to traffic flow and traffic jam modeling. Oper Res 51(5):681–710
28. Neumann JV (1966) Theory of self-reproducing automata. University of Illinois Press, Champaign
29. Nishinari K, Fukui M, Schadschneider A (2004) A stochastic cellular automaton model for traffic flow with multiple metastable states. J Phys A, Math Gen 37(9):3101
30. Prigogine I, Andrews FC (1960) A Boltzmann-like approach for traffic flow. Oper Res 8:789–797
31. Rahman SM, Ratrout NT (2009) Review of the fuzzy logic based approach in traffic signal control: prospects in Saudi Arabia. J Transp Syst Eng Inf Technol 9(5):58–70
32. Schadschneider A (2000) Statistical physics of traffic flow. Phys A, Stat Mech Appl 285(1–2):101–120
33. Schadschneider A (2002) Traffic flow: a statistical physics point of view. Phys A, Stat Mech Appl 313(1–2):153–187
34. Schreckenberg M, Schadschneider A, Nagel K, Ito N (1995) Discrete stochastic models for traffic flow. Phys Rev E 51:2939–2949
35. Sirakoulis GC (2004) A tcad system for vlsi implementation of the cvd process using vhdl. Integr VLSI J 37(1):63–81
36. Sirakoulis GC, Bandini S (eds) (2012) Cellular automata—proceedings of the 10th international conference on cellular automata for research and industry, ACRI 2012, Santorini Island, Greece, 24–27 September 2012. Lecture notes in computer science, vol 7495. Springer, Berlin
37. Sirakoulis GC, Karafyllidis I, Mardiris V, Thanailakis A (1999) Study of lithography profiles developed on non-planar Si surfaces. Nanotechnology 10(4):421
38. Sirakoulis GC, Karafyllidis I, Thanailakis A (2003) A cad system for the construction and vlsi implementation of cellular automata algorithms using vhdl. Microprocess Microsyst 27(8):381–396
39. Sirakoulis GC, Karafyllidis I, Spataro W (2009) A computational intelligent oxidation process model and its vlsi implementation. In: Proceedings of the 2009 international conference on scientific computing, CSC 2009, 13–16 July 2009, CSREA Press, Las Vegas, pp 329–335
40. Spataro W, Avolio MV, Lupiano V, Trunfio GA, Rongo R, D'Ambrosio D (2010) The latest release of the lava flows simulation model sciara: first application to mt etna (Italy) and solution of the anisotropic flow direction problem on an ideal surface. Proc Comput Sci 1(1):17–26
41. Spezzano G, Talia D, Gregorio SD, Rongo R, Spataro W (1996) A parallel cellular tool for interactive modeling and simulation. Comput Sci Eng 3:33–43
42. Szklarski J (2010) Cellular automata model of self-organizing traffic control in urban networks. Bull Pol Acad Sci, Tech Sci 58(3):435–441
43. Takayasu M, Takayasu H (1993) 1/f noise in a traffic model. Fractals 1(4):860–866
44. Toffoli T (1984) Cam: a high-performance cellular-automaton machine. Phys D, Nonlinear Phenom 10(1–2):195–204
45. Tripp JL, Mortveit HS, Gokhale M (2004) Acceleration of traffic simulation on reconfigurable hardware. In: Proceedings of the 2004 MAPLD international conference
46. Varas A, Cornejo MD, Toledo BA, Muñoz V, Rogan J, Zarama R, Valdivia JA (2009) Resonance, criticality, and emergence in city traffic investigated in cellular automaton models. Phys Rev E 80:056108
47. Wei J, Wang A, Du N (2005) Study of self-organizing control of traffic signals in an urban network based on cellular automata. IEEE Trans Veh Technol 54(2):744–748
48. Wilding N, Trew A, Hawick K, Pawley G (1991) Scientific modeling with massively parallel simd computers. Proc IEEE 79(4):574–585
49. Wolf DE (1999) Cellular automata for traffic simulations. Phys A, Stat Mech Appl 263(1–4):438–451