# Cellular automata and cluster computing: An application to the simulation of laser dynamics

J.L. Guisado
*Centro Universitario de Mérida, Universidad de Extremadura,*
*Santa Teresa Jornet, 38. 06800 Mérida (Badajoz), Spain.*
*jlguisado@unex.es*

F. Jiménez-Morales
*Departamento de Física de la Materia Condensada, Universidad de Sevilla,*
*P.O.Box 1065, 41080 Sevilla, Spain.*

F. Fernández de Vega
*Centro Universitario de Mérida, Universidad de Extremadura,*
*Sta. Teresa Jornet, 38. 06800 Mérida (Badajoz), Spain.*

Firstly, the application of a cellular automata (CA) model to simulate the dynamics of lasers is reviewed. With this kind of model, the macroscopic properties of the laser system emerge as a cooperative phenomenon from elementary components locally inter-acting under simple rules. Secondly, a parallel implementation of this kind of model for distributed-memory parallel computers is presented. Performance and scalability of this parallel implementation running on a computer cluster are analyzed, giving very satisfac-tory results. This confirms the feasibility of running large 3D simulations—unaffordable on an individual machine—on computer clusters, in order to simulate specific real laser systems.

*Keywords*: Parallel computing; modelling and simulation; computer clusters; complex systems; cellular automata; laser physics.

## 1. Introduction

Cellular automata (CA) are a class of spatially and temporally discrete systems which are characterized by local interaction and synchronous dynamical evolution [1, 2, 3]. Formally, a cellular automaton is a 4-tuple of the form [4]:

$$A = (\mathcal{L}, \mathcal{S}, \mathcal{N}, g)$$

where:

- $\mathcal{L}$ is the cellular space, usually a regular lattice, whose elements are called cells;
- $\mathcal{S}$ is a finite set of states;

- $\mathcal{N}$ is a neighborhood, i.e. a finite set (of size $\sigma = |\mathcal{N}|$) of neighborhood indices that identify the cells that are considered as the neighbors of a generic cell.
- $g$ is a local transition function which associates a new state to each state configurations in the neighborhood, $g: S^\sigma \to \mathcal{S}$.

The dynamics of the automaton is given by the synchronous application of the local function to all the cells in the cellular space. Thus, the state of each cell at any time step is determined as a function of the states of the neighboring cells at the previous time step.

CA are capable of generating a complex behavior emerging from sets of components which interact locally using relatively simple rules. Therefore, CA provide an excellent modeling and simulation approach for complex systems. CA models have been extensively used in the last decades to build computational models and simulations in many areas of science and technology [5, 6]. CA are intrinsically parallel systems. Therefore, they represent a method which can be naturally and efficiently implemented on parallel computers. They can be used to carry out high performance simulations taking full advantage of all the potential of parallel computing platforms. For that reason, and for the affordability of clusters of workstations with a very good price/performance ratio, parallel CA simulations have been successfully applied recently in many scientific and engineering fields, see for example [7, 8, 9, 10]. In addition, several software tools for the programming of CA on parallel computers (for example [11, 12]) have been introduced.

In this paper, CA are employed to model the dynamics of laser systems. A laser is a device that generates and amplifies coherent electromagnetic radiation based on the stimulated emission phenomenon. They are considered as paradigmatic complex systems, in which the interactions among simple atoms and the radiation that they produce can give rise to a cooperative phenomenon [13]. The four-level laser system shown in Fig. 1 is a simplified but still realistic description of many real laser systems. Electrons in the laser active medium are excited from its ground quantum-mechanical energy level ($E_0$) to a higher level ($E_3$) by the energy supplied by some external (electrical, optical or chemical) *pumping process.* In a good laser system, the life times of energy levels $E_3$ and $E_1$ are negligible as compared to the life time of level $E_2$. Thus, after being excited, electrons in level $E_3$ decay very fast down to level $E_2$, which has a longer life time. Electrons in level $E_1$ decay very fast down to level $E_0$ as well, so a population inversion is produced between levels $E_1$ and $E_2$: level $E_2$, which has a higher energy, is more populated than level $E_1$. The population inversion $N(t)$ is defined as the difference in the population (or number of electrons) in these energy levels: $N(t) = N_2(t) - N_1(t)$ where $N_i$ is the population of level $i$. As $N_1 \simeq 0$, the population inversion is approximately equal to the population of level $E_2$: $N(t) = N_2(t) - N_1(t) \simeq N_2(t)$.

*Stimulated emision* is the main process which is responsible of laser action: an electron in level $E_2$ (also referred to as the *higher laser level*) can decay down to

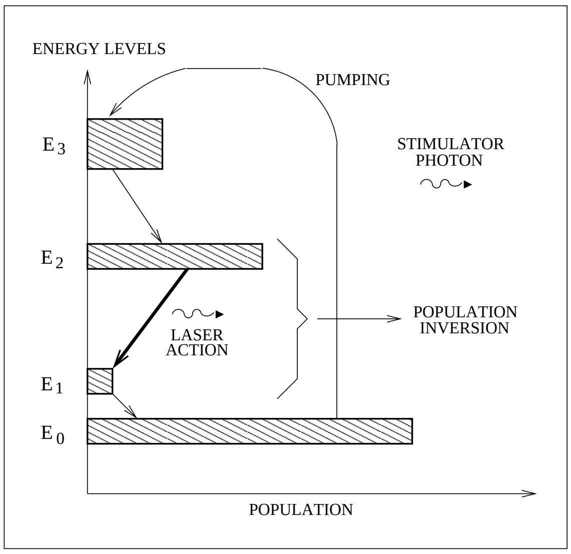


Fig. 1. Schematic view of a four-level laser system—a simplified but still realistic description of many real laser systems—showing some of its physical processes. Stimulated emission occurs when an electron in level $E_2$ decays down to level $E_1$ stimulated by the presence of a stimulator photon with energy $E = E_2 - E_1$. Some external pumping process excites electrons from the ground level up to level $E_3$. Population inversion is produced between levels $E_1$ and $E_2$, because the life times of energy levels $E_3$ and $E_1$ are negligible as compared to the life time of level $E_2$, so electrons in levels $E_3$ and $E_1$ decay very fast but level $E_2$ is metastable. Absorption of electrons in level $E_2$ and absorption of laser photons, which are also important, are not represented here.

level $E_1$ (also referred to as the *lower laser level*) stimulated by the presence of a photon with energy equal to the difference between the two energy levels. At the same time, a second photon with the same properties as the original one (wavelength, phase, polarization and direction of propagation) is emitted. In addition to pumping and stimulated emission, two other important processes occur in the laser system: *absorption of electrons* and *absorption of laser photons*. Absorption of electons happens when electons in the upper laser level ($E_2$) decay down to lower levels by other processes different than stimulated emission. Absorption of laser photons happens when they disappear from the laser device, either because they leave the cavity or because they are absorved by the laser materials. Population inversion is essential for laser operation because when it happens stimulated emision has a greater probability than electron absorption and the laser action can be sustained.

The standard description of laser dynamics is based on differential equations. They can be put in their simplest form as a system of two coupled differential equations (*laser rate equations*) [14, 15]: one of them giving the temporal variation of the total number of laser photons in the system $n(t)$ and the other one the temporal variation of the total population inversion $N(t)$:

$$\frac{dn(t)}{dt} = K\,N(t)\,n(t) - \frac{n(t)}{\tau_c} \tag{1}$$

$$\frac{dN(t)}{dt} = R - \frac{N(t)}{\tau_a} - K\,N(t)\,n(t) \tag{2}$$

Equation (1) gives the variation on the number of laser photons $n(t)$ with time, which is proportional to the laser beam intensity. The first term on its right hand side $+KN(t)n(t)$ represents the increase on the number of photons by stimulated emission ($K$ is the coupling constant between the radiation and the population inversion). The second term $-n(t)/\tau_c$ reflects the decaying (or absorption) process of laser photons inside the cavity with a characteristic decay time $\tau_c$. Eq. (2) represents the temporal variation of the population inversion $N(t)$. The term $+R(t)$ introduces the pumping of electrons with a pumping rate $R$ to the upper laser level. The term $-N(t)/\tau_a$ represents the decaying (or absorption) process of the population inversion, i.e. the decaying of electrons from the upper laser level to lower levels, with a characteristic decay time $\tau_a$. Finally, the product term $-KN(t)n(t)$ accounts for the decreasing of the population inversion by stimulated emission. These two coupled equations have a nonlinear nature due to the presence of the product term $KN(t)n(t)$ in each equation. In the case of small amplitude fluctuations its solutions can exhibit relaxation oscillations in their evolution towards a steady state. Furthermore, in the case of strong oscillations there do not seem to be any simple analytic solution. In this case, the two variables $n(t)$ and $N(t)$ are changing in a fast and typically nonlinear way [16, 14].

In the first part of this work we review the application of a model of laser dynamics based on CA, introduced in references [16, 17, 18], which can reproduce much of the phenomenology observed in laser systems. In the second part of this work, firstly we present a parallel implementation of the CA laser model for distributed-memory parallel computers. Secondly, we describe its implementation on a computer cluster using the message passing paradigm. Thirdly, an analysis of the performance and scalability of the parallel implementation running on the cluster is performed. This implementation will allow us to run large size two-dimensional (2D) simulations of the model on clusters of workstations. Additionally the 2D implementation will be useful to test the feasibility of a parallel three-dimensional (3D) version of the model, needed to make realistic simulations of specific laser systems. A parallel implementation is essential for the 3D version due to its high runtime and memory requirements.

The outline of the paper is the following: in section 2 the cellular automata model is presented; in Section 3 the results of the simulations are reviewed; in Section 4 a parallel implementation of the CA model is described; in Section 5 performance and scalability of the parallel implementation running on a computer cluster is analyzed; finally, the conclusions are summarized in Section 6.

## 2. Model of laser dynamics

In this section, the CA model for laser dynamics, originally introduced in reference [16], is presented. A laser device is modeled by a two-dimensional, multivariable, partially probabilistic CA, wich corresponds to a transverse section of the active medium in the laser cavity. The CA is defined by the following elements:

### 2.1. *Cellular space*

The CA is formed by a two-dimensional square lattice containing $N_c = L \times L$ cells, with periodic boundary conditions. It means that there is no border present in the lattice, what simulates an infinite lattice.

### 2.2. *States of the cells*

There are two variables, $a_i(t)$ and $c_i(t)$, associated to each cell. The first variable, $a_i(t)$, represents the state of the electron in cell $i$ at time $t$: if $a_i(t) = 0$ the electron is in the laser ground state and if $a_i(t) = 1$ it is in the upper laser state. The second variable, $c_i(t) \in \{0, 1, 2, ..., M\}$, represents the number of laser photons in cell $i$ at time $t$. The upper value $M$ is chosen large enough to avoid saturation of the system. The state variables values, which represent "bunches" of real photons and electrons, are obviously smaller than the number of photons and electrons in the real system and are connected to them by a normalization constant.

### 2.3. *Neighborhood*

Each cell interacts locally with nine neighboring cells included in its *Moore neighborhood*: the cell itself, its four nearest neighbors and the four next nearest neighbors. The Moore neighborhood in a rectangular or square lattice is anisotropic (i.e. doesn't have the same connectivity in every direction). However, when using probabilistic rules all directions are equally likely on a macroscopic level. Therefore, a simmetry can emerge on a macroscopic scale that was not present at the microscopic level [6]. The implementation of the model with this simple kind of lattice is accurate enough to reproduce the laser phenomenology. If a more detailed reproduction would be needed, a hexagonal lattice (which is isotropic) could be used.

### 2.4. *Transition rules*

The set of transition rules defines the time evolution of the CA, by determining the state of each cell of the system at time step t+1 depending on the state of the cells included in its neighborhood at time t. These rules represent different physical processes working at the microscopic level in a laser system:

- Rule 1. Pumping: If the electronic state of a cell has a value of $a_i(t) = 0$ in time $t$, then in time $t + 1$ that state will have a value of $a_i(t + 1) = 1$ with a probability pumping $\lambda$.

- Rule 2. Stimulated emission: If, in time $t$, the electronic state of a cell has a value of $a_i(t) = 1$ and the sum of the values of the laser photons states in the nine neighbor cells is greater than a certain threshold (which in our simulations has been taken to be 1), then in time $t + 1$ a new photon will be emitted in that cell: $c_i(t + 1) = c_i(t) + 1$ and the electron will decay to the ground level: $a_i(t + 1) = 0$.
- Rule 3. Photon decay: A finite life time $\tau_c$ is assigned to each photon when it is created. The number of time steps elapsed since the emission of each photon is individually recorded and each photon is destroyed $\tau_c$ time steps after being created.
- Rule 4. Electron decay: A finite life time $\tau_a$ is assigned to each electron that is promoted from the ground level to the upper laser level. The number of time steps elapsed since the excitation of each electron is individually recorded and each electron decays to the ground level again $\tau_a$ time steps after being excited, if it has not yet decayed by stimulated emission. The $\tau_a$ lifetime include spontaneous radiative and eventually non-radiative processes. As in an ideal four level laser the population of the lower laser level is negligible, stimulated absorption has not been considered.

Spontaneous emission and thermal contributions are simulated by a continuous noise of random photons in the laser mode which are introduced at every time step. They are responsible of the initial laser start-up, as happens in real lasers. This is done by making $c_i(t + 1) = c_i(t) + 1$ for a small number of randomly chosen cells ($< 0.01\%$ of total).

The response of the system is dependent on three parameters: the pumping probability ($\lambda$), the life time of photons ($\tau_c$) and the life time of excited electrons ($\tau_a$). For each simulation, an initial state is provided ( $a_i(0) = 0,\ c_i(0) = 0,\ \forall i$, except a small fraction 0.01% of noise photons present) and the system is let to evolve for a number of time steps. In each step, two macroscopic magnitudes are measured: the total number of laser photons $n(t) = \sum_{i=1}^{N_c} \mathrm{c}_{\mathrm{i}}(t)$, and the total number of electrons in the upper laser state (population inversion) $N(t) = \sum_{i=1}^{N_c} a_{\mathrm{i}}(t)$.

## 3. Simulation results

In this section, some of the results that have been previously obtained using the CA model of laser dynamics introduced in section 2, which were presented in references [16, 19, 17], are reviewed. As is shown, this simple model can reproduce, in a qualitative level, different physical features of a laser system.

First of all, the model can reproduce the typical laser behavior consisting in having a threshold pumping probability [16]. In a laser system, the laser mechanism only acts when the pumping probability is over a characteristic threshold value. Below this threshold, population inversion is not strong enough to trigger the laser action. This phenomenon is reproduced by the CA model: in the simulations, a cascade of laser photons are produced via stimulated emission only when the pumping probability is over a threshold value. This value depends on the other two system parameters (life times $\tau_a$ and $\tau_c$) and its dependence is in good agreement with the laser behavior as shown in Fig. 2.

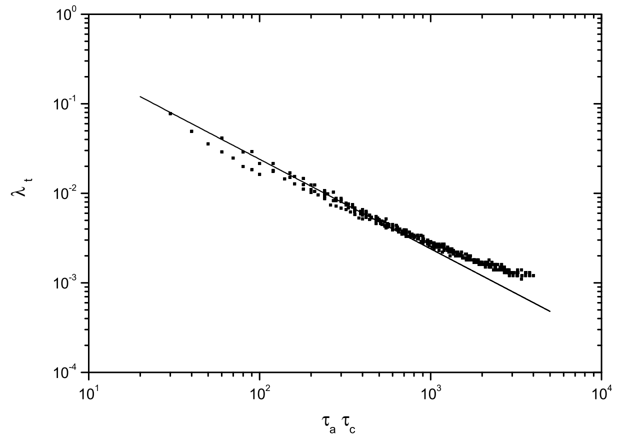


Fig. 2. Dependence of the threshold pumping probability $\lambda_t$ from the CA laser model on the product of the characteristic life times $\tau_a$ and $\tau_c$ (measured in time steps) plotted on a logarithmic scale. The dots correspond to the results obtained by the simulations and the solid line to the laser behavior, which is predicted by the standard laser rate equations, i.e. Eqs. (1-2).

Different types of behavior are shown in the time evolution of the macroscopic variables of the system—the total number of laser photons ($n$) and the population inversion ($N$)—depending on the value of the three system parameters. It is possible to classify the occurrence of each type of behavior in the parametric space by calculating the Shannon's entropy $S$ of the distribution of values achieved by $n$ or $N$, after running the simulation for a time interval [19]. In order to calculate $S$, the whole range of values taken by $n$ or $N$ is divided into $10^3$ equally spaced bins and the frequency ($f_i$) at which $n$ or $N$ has a value which is inside every particular non-void bin $i$ is computed. Then, the Shannon's entropy is:

$$S(\lambda, \tau_c, \tau_a) = -\sum_{i=1}^{m} f_i \ \log_2 f_i \tag{3}$$

where $m <= 10^3$ is the number of non-void bins. The value of $S$ is zero if the outcome of the system is constant, so that all the bins except one are void, and it increases as the probability distribution of values of the outcome is wider.

The result is shown in Fig. 3, where $R$ is the laser pumping rate and $R_t$ is the threshold laser pumping rate, which are linearly related to the pumping probability

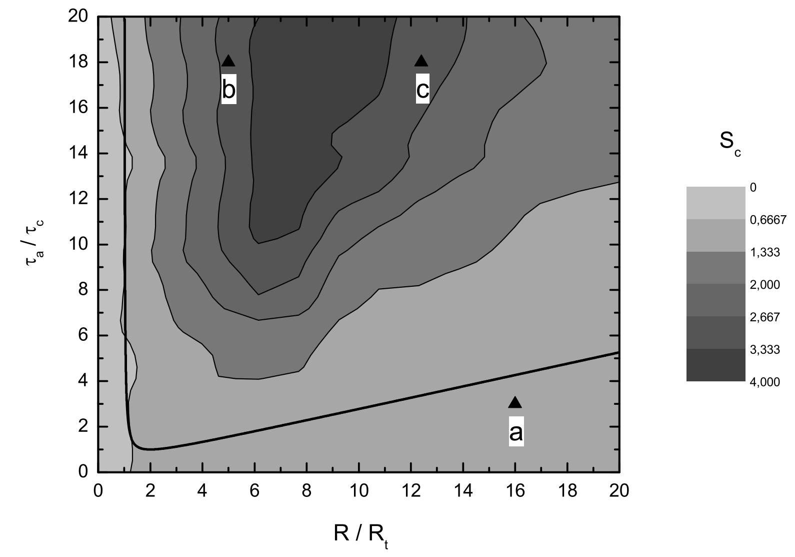


Fig. 3. Contour plot of the Shannon's entropy of the distribution of the number of laser photons obtained in the simulations for a fixed value of $\tau_c = 10$ time steps. High values of $S_c$ (dark zones) indicate that the response of the system is oscillatory, while low values (bright zones) correspond to a non-oscillatory response. The black line is the theoretical stability curve that indicates the predictions of the laser rate equations, separating areas of oscillatory behavior (above and to the right of the curve) and constant behavior (below and to the left of it). Points $a$, $b$ and $c$ correspond to the values of the parameters for which the time evolution is shown in Figs. 4 and 5. a: $R/R_t = 16$, $\tau_a/\tau_c = 3$. b: $R/R_t = 5$, $\tau_a/\tau_c = 18$. c: $R/R_t = 12.4$, $\tau_a/\tau_c = 18$. A 200 $\times$ 200 lattice has been used for this figure.

$\lambda$ and the threshold pumping probability $\lambda_t$ that appear in the CA model, so that $\frac{R}{R_t} = \frac{\lambda}{\lambda_t}$. The threshold value $\lambda_t$ is calculated as the smallest value of the pumping probability $\lambda$ for which after a transient time the number of laser photons is clearly greater than the number of noise photons introduced. Two main characteristic types of behavior are exhibited by the system: a stationary constant behavior for values of the parameters for which the Shannon's entropy $S_c$ is low, such as point $a$ in Fig. 3; and a regime with correlated large amplitude damped oscillations in the number of laser photons and the population inversion when $S_c$ is higher, such as point $b$. The later behavior is also known in laser physics as laser relaxation oscillations or laser spiking. In Fig. 4, the time evolution of the system for values of the parameters characteristic of these two regimes is shown.

This result can be compared with the predictions of the usual modelling approach based on the laser rate equations, Eqs. (1)-(2). By linearizing them for the case of small amplitude fluctuations, it is found that two different behaviors are expected, depending on the values of the laser parameters [16]: Damped oscillations

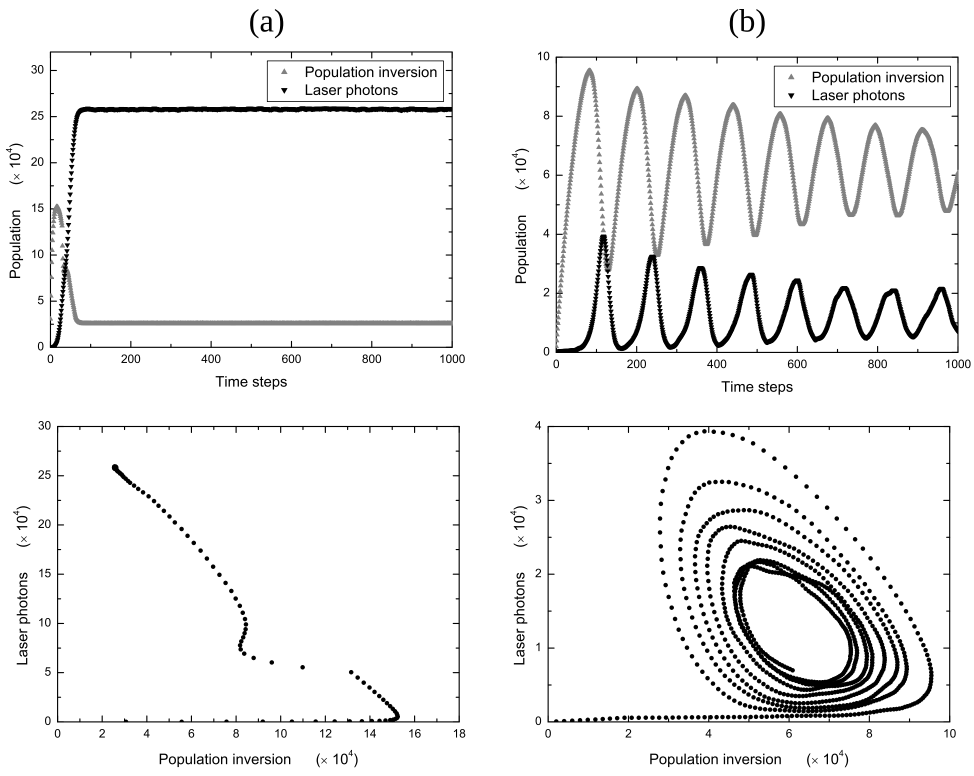


Fig. 4. (a): Evolution of the system for the values of the parameters marked as point $a$ in Fig. 3. Up: number of laser photons $(n(t))$ and population inversion $(N(t))$ versus time. Down: evolution in a phase space with the number of laser photons $(n(t))$ versus the population inversion $(N(t))$. After an initial transient, the systems goes to a fixed point. Parameters: $\lambda = 0.192$, $\tau_c = 10$, $\tau_a = 30$. (b): Evolution of the system for the values of the parameters marked as point $b$ in Fig. 3. The system follows a spiral trajectory which converges toward a steady-state limit point. Parameters: $\lambda = 0.0125$, $\tau_c = 10$, $\tau_a = 180$. Lattice size: $400 \times 400$ cells.

appear for values of the parameters obeying

$$\frac{\tau_a}{\tau_c} > \frac{\left(\frac{R}{R_t}\right)^2}{4\left(\frac{R}{R_t} - 1\right)} \tag{4}$$

and a constant behavior appears if this condition is not satisfied. The black line in Fig. 3 is the theoretical stability curve derived from equation (4), which separates the area of damped oscillations (above and to the right of the curve) and the area of a constant behavior (below and to the left of it) in the parametric space. As seen in Fig. 3, the dependence of the type of behavior exhibited by the simulations (classified by the Shannon's entropy) on the parameters is in a good qualitative agreement with the theoretical stability curve.

Besides the two main types of characteristic behavior presented above, the CA

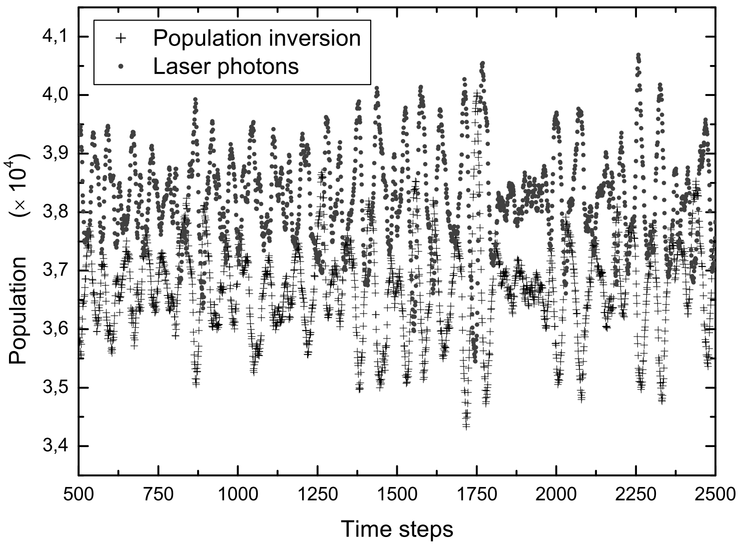


Fig. 5. Evolution of the system follows an irregular oscillations regime for the values of the parameters marked as point $c$ in Fig. 3. The number of laser photons and population inversion versus time is shown, after a transient of 500 time steps. Fluctuations are exhibited on a wide range of time scales. Parameters: $\lambda = 0.031$, $\tau_c = 10$, $\tau_a = 180$. Lattice size: 400 $\times$ 400 cells.

model displays another type of complex behavior [17], which is shown in Fig. 5 and corresponds to the values of the parameters of the point $c$ in Fig. 3. After a transient time, the system exhibits irregular oscillations involving fluctuations on a wide range of time scales. In order to distinguish these oscillations from those that could be induced by the introduction of noise photons, in this case noise photons have only been introduced in the first 100 time steps to activate the laser process, and the populations values have been recorded after a transient time of 500 time steps. The power spectrum of a time series is a useful tool to analyse its statistical behavior. The power spectrum of the time series $\{x_j \equiv x(t_j)\}$, where time is sampled as $t_j \equiv j\,\Delta t$ with $j = 1, 2, ..., N$ and frequency $f$ is defined as the inverse of time $f \equiv t_j^{-1}$, is the square of the modulus of the Fourier transform $(\hat{x}_k)$ of the series:

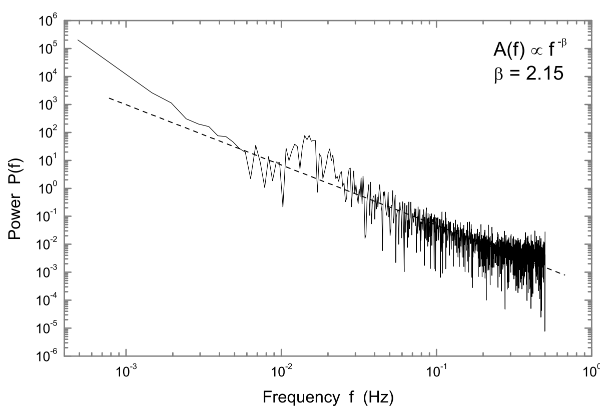
$$P(f_k) = |\hat{x}_k|^2$$

where

$$\hat{x}_j = \sum_{k=1}^{N} x_k e^{i2\pi k(j/N)}$$

.

Figure 6 shows the corresponding power spectrum $P(f)$, which follows a power law of the kind $1/f^{-\beta}$ where $f$ is the frequency and the exponent is close to $\beta = 2$. The system could be in a chaotic state for this regime of irregular oscillations, which



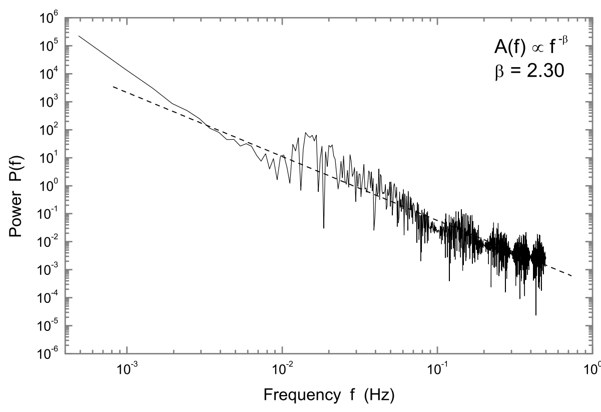


Fig. 6. Power spectrum $P(f)$ with respect to the frequency $f$ of the population inversion (up) and the number of laser photons (down) for the time series depicted in Fig. 5, showing a power law behavior. A linear fit of the spectrum, $A(f)$ (dotted line), varies as $f^{-\beta}$ with exponent $\beta = 2.15$ for the population inversion and $\beta = 2.30$ for the number of laser photons.

would be in agreement with the frequent finding of a chaotic dynamics in real lasers for sufficiently high values of the pumping. A process of self-organized criticality could also be the origin of the observed behavior. However, a detailed study of the dynamics of the system in this regime has not been yet fully accomplished.

Other aspects of the laser phenomenology have also been successfully modeled using this kind of CA model, such as the dynamics of pulsed pumped lasers [18], and the dependence of the decay rate of the relaxation oscillations on different parameters [17]. In addition, CA models of laser dynamics can also be very useful to study the evolution of spatiotemporal patterns in lasers.

## 4. Parallel implementation

In the previous section, different results obtained with the CA model of laser dynamics have been reviewed, showing that a very simple, coarse-grained CA model can reproduce the laser behavior qualitatively. However, if one is interested in reproducing the behavior of some specific laser device quantitatively, a more fine-grained model is needed which can reproduce more details of that particular device (for example, complicated boundary conditions) and have a granularity which is closer to the real macroscopic system. In addition, a 3D version of the CA model is needed to reproduce the shape of the device. For both purposes (using a fine-grained or a 3D CA model), a very large lattice size is needed. The runtime for a typical experiment grows very quickly with the automaton size, as shown in Fig. 7, so an extra large runtime would be needed on a stand-alone sequential computer. Therefore, a parallel implementation is mandatory for this kind of applications.

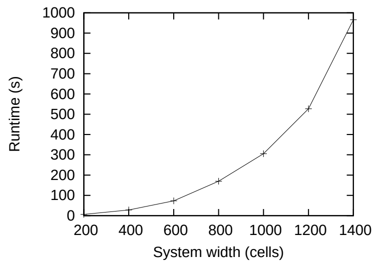


Fig. 7. Variation of the runtime of a sequential implementation of the 2D laser model with the system size. For all the experiments, the simulation has been run for 1000 time steps using the same values of the parameters.

We have developed a parallel implementation of the CA model of laser dynamics for parallel computers with distributed memory, such as computer clusters, using the message passing paradigm. In this paradigm a set of processes which have only local memory operate on different processors. Processes can perform cooperative operations by communicating among themselves by sending and receiving messages which can include data to be exchanged and control information. The two most frequently used implementations of the message passing paradigm are *Message Passing Interface (MPI)* and *Parallel Virtual Machine (PVM)*. Both are public domain libraries that can target very different types of machines and that can be used with different programming languages. While MPI is currently more extended, we have used PVM for the implementation of the CA model of laser dynamics because we are interested in a further study of that model using dynamic load balancing mechanisms specifically developed for this library. However, the implementation can be easily ported to other message passing libraries such as MPI.

The *master-slave* programming model has been used for the parallelization. The procedure is illustrated in Fig. 8. Workload allocation has been carried out following the *data decomposition* or *partitioning* methodology where identical tasks operate over different portions of the data.

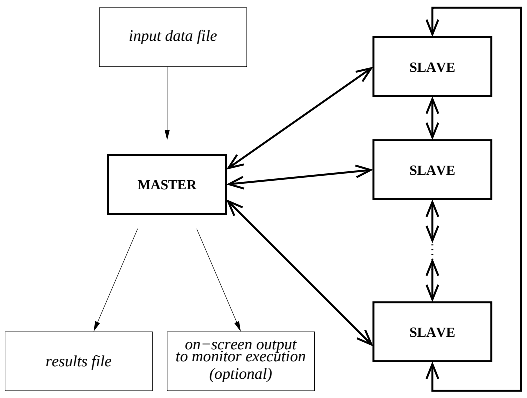


Fig. 8. Block diagram of the parallel implementation of the CA model of laser dynamics. The master-slave programming model has been used. The master program performs initialization, partitioning of the problem, distribution of parts of the task to the slave programs, collection of results and input/output. A slave program is executed on each cluster node and performs the computation for its assigned partition of the CA. Each box in bold type represents a different processor, bold lines represent communications between the processes running on different processors and arrows indicate direction of communication or data flow.

The "*master program*" initially divides the CA lattice in $p$ partitions of equal size and sends each to a "*slave program*" running on a different processor. The particular tasks performed by the master and slave programs are:

- Master program:
  1. Input data from external file (system size, number of partitions, parameter values, number of time steps) and initialization.
  2. Spawning of slave programs.
  3. Partitioning of the initial data of the automaton.
  4. Sending of common information and initial data to each slave.
  5. Collection of results from slaves for each time step.
  6. Termination of slave programs.
  7. Calculations performed using collected data.
  8. Output of final data to external files.
  9. Timing functions to measure performance.
- Slave program:
  1. Reception of common information and initial data from master.
  2. Time evolution computation for the assigned partition: application of CA evolution rules.

(3) Exchange of state of the boundary cells with slave programs computing the neighboring partitions.
(4) Computation of intermediate results and their communication to master program.

In general, for d-dimensional CA 1-, 2-, ... or d-dimensional domain decompositions can be considered. In the present model, a 2D CA is used so there are two possibilities: using a one-dimensional (1D) domain decomposition, i.e. partitioning the CA into parallel stripes, or using a 2D domain decomposition into a checkerboard pattern. Both alternatives are shown in Fig. 9.

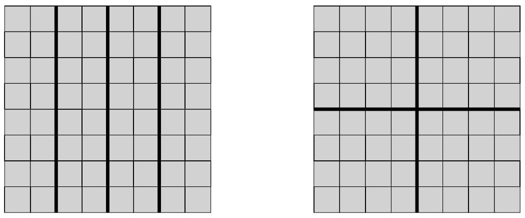


Fig. 9. Posible data domain decompositions for a 2D CA. Left: 1D domain decomposition into parallel stripes. Right: 2D domain decomposition into a checkerboard pattern. Each processor of the parallel computer performs computations on one subdomain. It is critical to minimize communications between different processors.

For a parallel implementation intended to be executed on a computer cluster, it is specially important to minimize the number of messages that exchange data between the processors, because communications have a higher time cost than computations on this kind of architecture. In order to exchange a message containing data, the sender process must call the *send* routine from the message passing library and the receiver process must call the *receive* routine. Therefore, to choose the most advantageous decomposition, it is convenient to compare the number of *send/receive* calls carried out by each subdomain for both alternatives. This value is shown in Table 1 for the possible types of data decomposition for a 2D CA, considering that for each communication 2 calls (one *send* and one *receive*) are used and that in the case of a 2D domain decomposition the number of send/receive calls per subdomain depends too on the type of neighborhood considered for the CA because a different number of surrounding cells from each subdomain must be communicated.

We have chosen a 1D domain decomposition (instead of a 2D decomposition) for the parallel implementation of the CA model of laser dynamics for the following reasons:

i) A 2D CA with Moore neighborhood is used, so the number of *send/receive* calls for a 1D domain decomposition is 4 times smaller than for a 2D decomposition, as shown in Table 1.

Table 1. Number of *send/receive* calls per subdomain, for different data domain decomposition types for a 2D cellular automaton.

| Domain decomposition | 1D | 2D (v-N) | 2D (M) |
|---|---|---|---|
| Number of calls | 4 | 8 | 16 |

*Note*:
(v-N): CA using the von Neumann neighborhood
(M): CA using the Moore neighborhood.

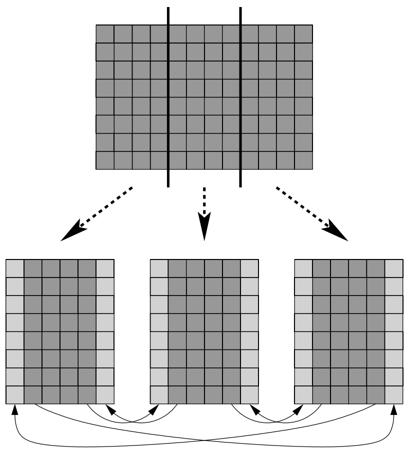


Fig. 10. The CA is vertically partitioned into parallel stripes (using a 1D domain decomposition) and each sub-domain is assigned to a different processor. Two additional columns—in light gray—of *ghost cells* are added at both sides of each partition to store the photon state $c_i(t)$ of neighboring cells belonging to the neighboring partitions.

ii) The communication structure is simpler and the data to be communicated can be stored in adjacent memory positions so that the joint access time can be reduced.
iii) As discussed in reference [20], a 1D decomposition is more favorable in runtime for a small to moderate number of nodes, despite the amount of data to be communicated is larger.

As shown in Fig. 10, the CA is vertically partitioned into parallel stripes and each sub-domain is assigned to a different processor. For each sub-domain, two additional columns of ghost cells have been included at the left and right sides, used to store the photon state $c_i(t)$ of neighboring cells belonging to different sub-domains.

Each slave program computes the time evolution on its assigned partition of the automaton by applying the CA evolution rules. To this end, the following procedures are applied on each time iteration:

(1) Stimulated emission.
(2) Refresh values of photon state $c_i(t)$.
(3) Photon and electron decay and electron pumping.
(4) Noise photons.

Procedures 1 to 3 are three successive loops through all of the cells in the partition. In the first one, stimulated emission is computed storing new values of the photon state $c_i(t)$ in a temporal array; in the second one, the values of the photon state $c_i(t)$ of all the cells in the partition are updated using the values stored in the temporal array; in the third one, photon and electron decay and electron pumping are computed. In the fourth procedure, $N_n/p$ noise photons are introduced in randomly chosen cells inside the partition, where $N_n$ is the total number of noise photons introduced in the system at every time step and $p$ is the number of partitions in the system.

In general, in a parallelized CA the state of the CA cells in the boundaries of each slave partition must be communicated to the slave programs dealing with the neighboring partitions, because this state will be needed to compute the CA evolution rules there. Synchronization is capital here: communication must be carried out first and computation afterwards, and not vice versa. For this particular CA model, the only state value from neighboring cells needed to compute the CA evolution rules for a cell is the photon state $c_i(t)$. Therefore, only this state is communicated to the neighboring partitions. This communication is carried out directly between the slave programs.

Additionally, in the last part of each time iteration, each slave program computes the total number of electrons $a_i(t)$ and laser photons $c_i(t)$ in its CA partition and sends this information to the master, which can record it and make some calculations with it, such as computing the Shannon's entropy as described in section 3.

## 5. Performance and scalability analysis

In order to test the performance and scalability of the parallel implementation of the 2D CA model of laser dynamics on small computer clusters, simulations have been carried out on a Linux PC cluster of ten nodes with Intel Pentium-4 processor, for different sizes of the cellular automaton lattice. The cluster is heterogeneous: six nodes have a clock frequency of 2.7 GHz and the other four nodes of 1.8 GHz. This configuration has been chosen for the experiments to avoid indeterminism in the results: for simulations with 1 to 6 nodes the slave programs have been run on the "fast" (2.7 GHz) machines, and for simulations with 7 to 10 nodes additional "slow" (1.8 GHz) machines have been used to achieve the required number of nodes. The master program was always run on the master node of the cluster (which runs at 1.8 GHz).

In Fig. 11, the results of the same experiment (computing the time evolution of the population inversion and the total number of laser photons for 1000 time steps) for two different system sizes are compared: $300 \times 300$ cells (a)—representing

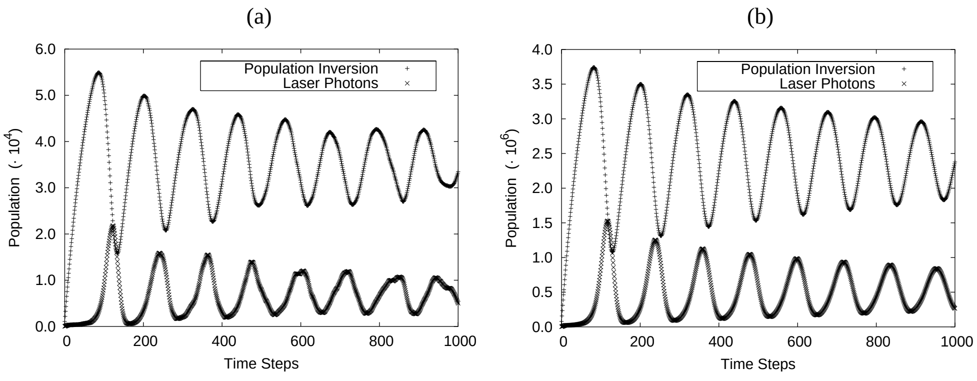


Fig. 11. Evolution of the whole system for the same values of the parameters (those marked as point $b$ in Fig. 3) for two different system sizes. (a) (left): 300 × 300 cells, typical result of previous sequential implementations of the model. (b) (right): 2520 × 2520 cells, using the parallel implementation with a much larger system size. Parameters: $\lambda = 0.0125$, $\tau_c = 10$, $\tau_a = 180$.

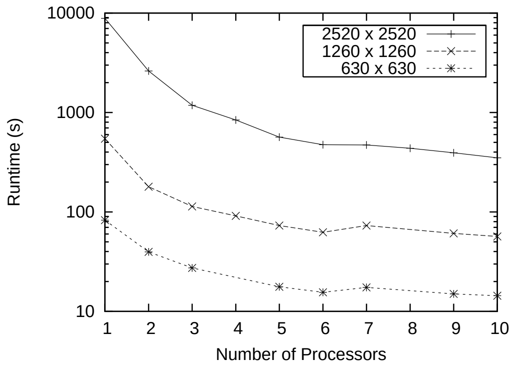


Fig. 12. Runtime of the experiments, shown in a logarithmic scale, using three different system sizes, for different number of partitions of the whole CA. Each partition was run on a different processor.

a typical system size for previous sequential implementations—and 2520 × 2520 cells (b). In both cases the same parameter values as in Fig. 4(b) have been used: $\lambda = 0.0125$, $\tau_c = 10$, $\tau_a = 180$. As for all the experiments included in this section, the ratio of noise photons (introduced in every time step) to total number of cells in the system has been maintained constant (0.03% of the cells) for the simulations with different system sizes. The differences between both results are that the populations are scaled and that the shape of the oscillations is smoother in (b) due to having used a much larger system size, thus reproducing more accurately the relaxation oscillations obtained in real lasers or from the integration of macroscopic differential equations.

The performance of the parallel implementation has been measured by running

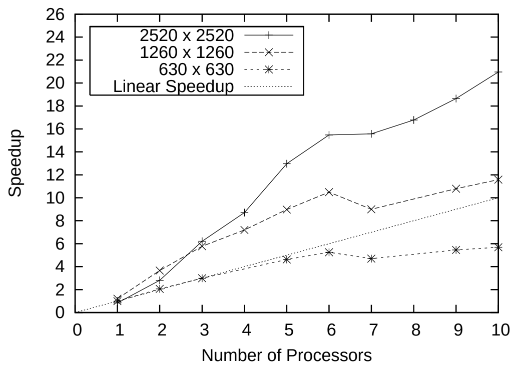


Fig. 13. Speedup of the parallel implementation with respect to the sequential program for varying number of processors and for three different system sizes, compared to the ideally optimal linear speedup. A very good performance is obtained for a moderate system size (630 × 630 cells). For larger system sizes, super-linear speedup is obtained.

the same experiment (corresponding to the results shown in Fig. 11) using different system sizes (2520 × 2520, 1260 × 1260 and 630 × 630 cells) and for different number of partitions of the whole CA (each one being handled by the slave program on a different node). Runtime of the experiments is shown in Fig. 12, using a logarithmic scale. Runtime get significantly shorter when the number of processors is increased, with the exception of the change from 6 to 7 processors. The reason is that "fast" machines have been used for a number of processors from 1 to 6, whereas "slow" machines are used to complete a number of processors higher than 6. As the CA operates in a lock-step mode, i.e. all the cells of the CA must be updated before proceeding to the next step, the processing speed of the application is limited by the speed of the slowest running task and adding one "slow" machine results in a decrease of speed of the whole application.

The performance of the parallel implementation can be evaluated in terms of the *speedup ($S_p$)*, which indicates how much faster is a parallel algorithm than a corresponding sequential algorithm. Speedup is defined [21] as the ratio of the runtime of the sequential version of the program running on 1 processor of the parallel computer ($T_1$) to the runtime of the parallel version running on $m$ processors of the same computer ($T_m$):

$$S_p(m) = \frac{T_1}{T_m} \tag{5}$$

The speedup obtained for different number of partitions of the system (each one running on a different processor) is shown in Fig. 13, in comparison with the *linear speedup*, represented by the line $y = x$, which could be defined as the ideally optimal speedup. A parallel application obtains linear speedup when, by using $m$ processors, the runtime is reduced $m$ times with respect to the runtime of the sequential version of the program running on 1 processor, i.e. when $S_p(m) = m$. As shown in in Fig. 13, a very good performance has been obtained. In fact, a *super-linear speedup* (speedup higher than linear) is obtained, specially for a system size of 2520 $\times$ 2520 cells, and also for 1260 $\times$ 1260 cells. The reasons for super-linear speedup are finite memory effects on the memory hyerarchy. For CA with a large lattice size, swap memory is used for the sequential version of the program and not for the parallel version with more than one processor because the total size of available physical memory is higher. As an access to swap memory is much slower than an access to physical memory, this causes an extra speedup in addition to the speedup coming from computation. Additionally, a similar but slighter effect is caused by the memory caches: when using more processors, the size of accumulated memory caches from different processors is larger so that more data fit into caches and can be accessed faster. Due to these effects a very high speedup, even higher than linear, can be obtained.

The use of swap memory for running a very large size simulation of the model (as necessary for 3D simulation) can make the calculation non-affordable on a single PC but feasible on a cluster, because the system is partitioned so that less memory is used in each node and no swap memory is needed.

The execution of the parallel application has been analysed using XPVM, a graphical console for PVM which provides a performance monitor and debugger [22]. In Fig. 14, a Gantt chart showing the different types of tasks executed for each node versus time is shown. As can be observed in the figure, after the master process allocates the initial CA state to all the slaves, computation is the dominant task for slave processes for the rest of the execution, and the master process mostly waits.

In Fig. 15, a detail of the tasks executed by each node and the messages transferred between different nodes versus time, once the computation phase has started, is shown. Two distinct periods can be recognized: computation and communication. In the computation period, the CA state of the next time step is calculated by applying the evolution rules in each partition. In the communication period, the photon state values of the cells in the borders of each partition are communicated to the slave nodes which handle the neighbouring partition, and the total number of electrons and photons in each slave partition are sent to the master node. As can be observed in Fig. 15, computation periods are much longer than communication periods. The average computation-to-communication ratio obtained for the parallel application for slave processes is of the order of 10. As discused above, on a computer cluster it is essential to keep the amount of communications to a minimum, so that the computation-to-communication ratio is high. A relatively high value has been obtained. This is an indication that the application takes advantage of the parallelization on the computer cluster.

Finally, a scalability analysis of the parallel application running on the computer cluster has been carried out. An application is said to be scalable if, when the number of processors and the problem size are increased by a factor of $x$, the running

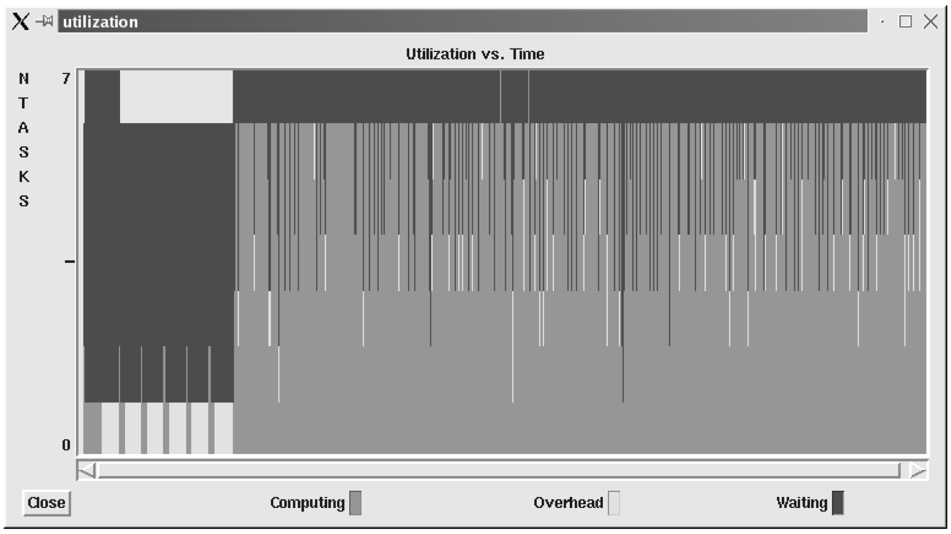
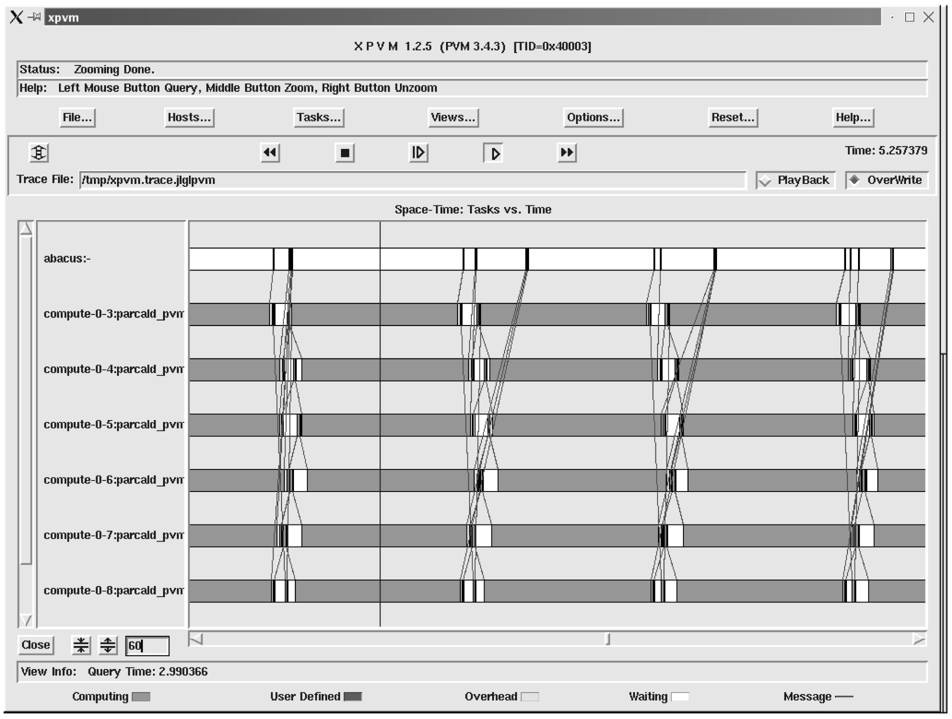


Fig. 14. Gantt chart showing the different tasks executed by each cluster node versus time. Tasks 1 to 6 (down) correspond to the slave nodes and task 7 (up) to the master node. The tasks marked as "Computing" are busy executing user computation, those marked as "Overhead" are busy executing PVM system calls and those marked as "Waiting" are idle and waiting for messages. Initially (left) the master process allocates the initial CA state to all the slaves. After this initialization phase, the slave processes spend most of the time computing and the master process mostly waits.

time remains the same [23] In order to analyze the scalability of our combination parallel application – parallel computer, the same experiment has been run increasing the total number of cells of the CA (which in our problem is directly related to the amount of computation, or problem size) and the number of processors by the same factor. The same experiment as for Fig. 11 has been used, but involving the computation of 10000 time steps. As in that case, in order to compute the same physical experiment, the ratio of noise photons to total number of cells in the system has been maintained constant (0.03% of the cells) for the simulations with different system sizes. The results are shown in Fig. 16. In an ideally scalable parallel computer, the running time should be the same in all cases. In practice, using a higher number of processors always involves some runtime overload due to the communication needed to coordinate and distribute the task between the processors. As shown in Fig. 11, only a small excess (from 2 % to 5 %) of runtime compared to the optimal value has been obtained, showing that the parallel implementation scales well on a small computer cluster.

## 6. Discusion and conclusions

In the first part of this work, the application of a cellular automata model simulating laser dynamics was reviewed. It was shown that laser dynamics phenomenology is reproduced by the CA model: the different laser behaviors and their dependence on


X P V M 1.2.5 (PVM 3.4.3) [TID=0x40003]

Status: Zooming Done.

Help: Left Mouse Button Query, Middle Button Zoom, Right Button Unzoom

Time: 5.257379

Trace File: /tmp/xpvm.trace.jlglpvm

View Info: Query Time: 2.990366

Fig. 15. Gantt chart showing a detail of the tasks executed by each cluster node and the messages passed between different nodes versus time, once the calculation phase has started. Computation periods are much longer than communication periods, so a high computation-to-communication ratio, of the order of 10, is obtained. That condition is of capital importance on a computer cluster and indicates that the application is exploiting the parallel computational power of the machine.

laser parameters are reproduced, as well as the threshold pumping probability for laser action and its dependence on laser parameters. CA-based models show some benefits over the standard approach based on differential equations for different situations. They can be used as a good modelling alternative for lasers governed by stiff differential equations, with convergence problems; or for very small optoelectronic devices, for which the usual approximations may not be valid. Also, boundary conditions with arbitrary geometry can be readily modelled. Another advantage is that CA-based models can be efficiently implemented on parallel computers thanks to their intrinsic parallel nature. In addition, we expect they could be useful to study problems of current interest, such as cooperative phenomena in lasers, chaotic lasers or two-photon lasers.

In the second part of this work, a parallelization of this model in two dimensions was studied, its implementation on a computer cluster using a message passing library was described, and the performance and scalability of the parallel implementation running on that parallel platform ware analyzed. Such an implementation is required in order to run realistic simulations, due to their extensive runtime and

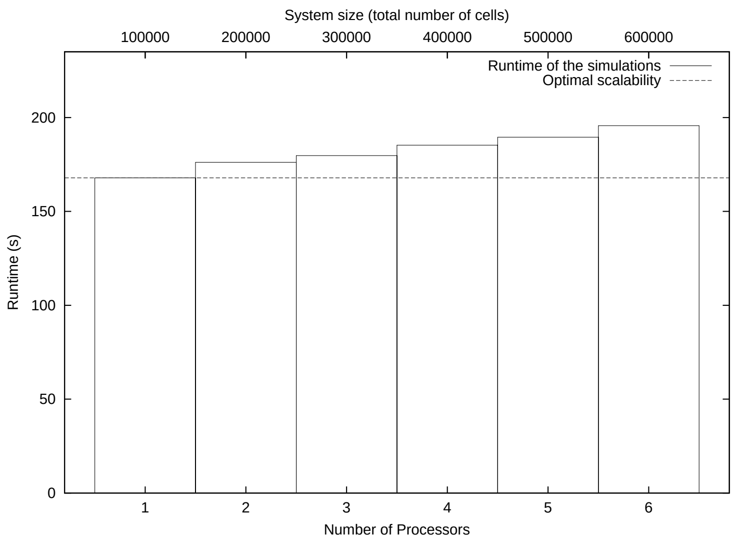


Fig. 16. Runtime of the parallel implementation of the application for different number of processors and system sizes, to show the scalability of the combination parallel application-parallel computer. For an optimal ideal scalability, the same runtime (horizontal straight line) would be obtained when increasing the number of processors and the system size by the same factor. A small excess in runtime from the optimal value is obtained, showing that the parallel implementation scales well at this level of parallelization.

memory size needs. In spite of the fact that the CA parallel model is only partially uncoupled, i.e. the parallel tasks have to perform communications after each time step, the presented implementation takes a good advantage of the parallelization and obtains a very good performance. The reason is that, after the initial data have been communicated to the slave processes and the computation phase has started, communication periods are much shorter than computation ones. Different factors contribute to that situation: it is not necessary to communicate the complete state of the neighboring cells to the neighboring processors, but just one of its variables ($c_i$); the chosen data domain decomposition (1D) minimizes the number of *send/receive* calls; the chosen parallel computer platform (a computer cluster) introduces only a minimum delay in the communications between processors (much smaller than parallel platforms of distributed architecture, such as in grid computing).

The present analysis confirms the feasibility of running large 3D CA simulations of specific real laser systems in computer clusters, which are not realizable on a single processor sequential computer due to the high runtime and memory requeriments.

## Acknowledgments

We thank the referee for his detailed suggestions for improvement. This work was partly supported by the project OPLINK (TIN2005-08818-C04-03) of Ministerio de Educación y Ciencia (Spain).

## References

[1] von Neumann, J., *Theory of Self-Reproducing Automata* (University of Illinois Press, Urbana, 1966).

[2] Wolfram, S., *Cellular automata and complexity* (Addison-Wesley, Reading, MA, 1994).

[3] Ilachinski, A., *Cellular automata. A discrete Universe* (World Scientific, Singapore, 2001).

[4] Weimar, J., *Simulation with Cellular Automata* (Logos Verlag, Berlin, 1998).

[5] Chopard, B. and Droz, M., *Cellular Automata Modeling of Physical Systems* (Cambridge University Press, Cambridge, 1998).

[6] Toffoli, T. and Margolus, N., *Cellular automata machines: a new environment for modelling* (The MIT Press, Cambridge, MA, 1987).

[7] Sloot, P., Kaandorp, J., Hoekstra, A. and Overeinder, B., Distributed simulation with cellular automata: architecture and applications. *Lecture Notes in Computer Science* **1725**, 203–248, 1999.

[8] Bandini, S., Mauri, G. and Serra, R., Cellular automata: from a theoretical parallel computational model to its application to complex systems. *Parallel Computing* **27**(5), 539–553, 2001.

[9] Dattilo, G. and Spezzano, G., Simulation of a cellular landslide model with CAMELOT on high performance computers. *Parallel Computing* **29**(10), 1403–1418, 2003.

[10] Zhou, J., Zhu, L., Petzold, L. and Yang, T., Parallel Simulation of Fluid Slip in a Microchannel. In *18th International Parallel and Distributed Processing Symposium (IPDPS'04) - Papers* (IEEE Computer Society, 2004), p. 43a.

[11] Talia, D., Cellular processing tools for high-performance simulation. *IEEE Computer* **33**(9), 44–52, 2000.

[12] Hecker, C., Roytenberg, D., Sack, J.-R. and Wang, Z., System development for parallel cellular automata and its applications. *Fut. Gen. Comp. Sys.* **16**, 235–247, 1999.

[13] Haken, H., *Synergetics: an introduction. Nonequilibrium phase transitions and self-organization in Physics, Chemistry and Biology* (Springer-Verlag, Berlin, 1983).

[14] Siegman, A., *Lasers* (Unversity Science Books, Mill Valley, CA, 1986).

[15] Svelto, O., *Principles of lasers* (Plenum Press, New York, 1998), 4th edition.

[16] Guisado, J. L., Jiménez-Morales, F. and Guerra, J. M., Cellular automaton model for the simulation of laser dynamics. *Physical Review E* **67**(6), 066708, 2003.

[17] Guisado, J. L., Jiménez-Morales, F. and Guerra, J. M., Computational simulation of laser dynamics as a cooperative phenomenon. *Physica Scripta* **T118**, 148–152, 2005.

[18] Guisado, J. L., Jiménez-Morales, F. and Guerra, J. M., Simulation of the dynamics of pulsed pumped lasers based on cellular automata. *Lecture Notes in Computer Science* **3305**, 278–285, 2004.

[19] Guisado, J. L., Jiménez-Morales, F. and Guerra, J. M., Application of Shannon's entropy to classify emergent behaviors in a simulation of laser dynamics. *Mathematical and Computer Modelling* **42**, 847–854, 2005.

[20] Worsch, T., Simulation of cellular automata. *Future Generation Computer Systems* **16**(2-3), 157–170, 1999.

[21] Foster, I., *Designing and building parallel programs* (Addison-Wesley, Reading, MA, 1995).

[22] XPVM: A Graphical Console and Monitor for PVM. http://www.netlib.org/utk/icl/xpvm/xpvm.html, as available on January 2007.

[23] Dongarra, J., Foster, I., Fox, G. C., Gropp, W., Kennedy, K., Torczon, L. and White, A. (eds.)., *Sourcebook of parallel computing* (Morgan Kaufmann, San Francisco, 2003).