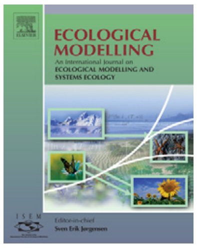
available at www.sciencedirect.com

journal homepage: www.elsevier.com/locate/ecolmodel



Modeling and simulation of fire spreading through the activity tracking paradigm

A. Muzy[a,*], J.J. Nutaro[b], B.P. Zeigler[c], P. Coquillard[d]

[a] Laboratory UMR CNRS LISA, Università di Corsica-Pasquale Paoli, UFR Drittu, Scenzi suciali, ecunòmichi è di gestioni, 22, av. Jean Nicoli, BP 52, 20250 Corti, France
[b] Oak Ridge National Laboratory, PO Box 2008, MS6085 Oak Ridge, USA
[c] Arizona Center for Integrative Modeling and Simulation, Department of Electrical and Computer Engineering, University of Arizona, 1230 E Speedway Boulevard, Tucson, AZ, USA
[d] Laboratory UMR Biotic Interactions and Plant Health, AgroBiotech Center, 400 route des Chappes, B.P.167, 06903 Valbone-Sophia Antipolis, France

ARTICLE INFO

Article history:
Received 19 February 2008
Received in revised form
25 July 2008
Accepted 22 August 2008
Published on line 17 October 2008

Keywords:
Fire spreading
Component-based modeling and simulation
Activity tracking paradigm

ABSTRACT

Modeling and simulation is essential for understanding complex ecological systems. However, knowledge of the structure and behavior of these systems is limited, and models must be revised frequently as our understanding of a system improves. Moreover, the dynamic, spatial distribution of activity in very large systems necessitates mapping natural mechanisms as logically as possible onto computer structures. This paper describes theoretical and algorithmic tools for building component-based models and simulations of dynamic spatial phenomena. These methods focus attention on and exploit the irregular distribution of activity in ecological processes. We use the DEVS formalism as the basis for a component-based approach to modeling spatially distributed systems. DEVS is a mathematical theory of discrete-event systems that is well suited for describing large systems that are described by small parts with irregular, short-range interactions. This event-based approach to modeling leads naturally to efficient simulations algorithms which focus on the active parts of a large model. Ecological modeling benefits from these efficient the simulation algorithms and the reusability of the model's basic components. Our event-based method is demonstrated with a physics-based model of fire spread.

1. Introduction

Ecosystem analysis and understanding mostly necessitates estimating values of parameters of active entities (animals or plants or abiotic components of the system). Thus, the greatest efforts of researchers are usually devoted to tracking activity (also called monitoring activity) in the components of ecosystems. Examples can be found in animal ecology (*e.g.*, radio and GPS tracking methods), behavioral ecology (*e.g.*, video tracking), plant ecology (*e.g.*, population dynamics, monitoring of growth, of seed dispersal, of chemical secretion and of climatic parameters) and in fresh water and marine ecosystems (*e.g.*, monitoring of fluxes, of hydrodynamics and of concentrations of various substances). On the contrary, little attention is paid to inactive entities, except from a descriptive point of view.

The ability to model ecological systems has been greatly improved over the last two decades by advances in the

* *Corresponding author.*
E-mail address: a.muzy@univ-corse.fr (A. Muzy).
0304-3800/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.ecolmodel.2008.08.017

methods used to build conceptual models and in the technology for simulating those models. Because ecology focuses on living entities, the paradigm of individual based modeling has been widely developed. Individual based models are implemented in several ways: deterministic cellular automata; stochastic cellular automata (Balzter et al., 1998); gaps models (forest growth models) in which each cell of a grid holds a list of entities with attributes and methods (Norby et al., 2001); multi-agents model mixing stochastic and deterministic processes (Parker et al., 2003) and multiple models in which the processes act on entities at several abstraction levels at the same time (from individuals to groups and to population) by means of various techniques (*e.g.*, Markov analysis, Monte Carlo simulations, Leslie's matrixes, EDO and EDP, and finite element calculations).

Object-oriented techniques are widely used for modeling and simulation of systems (Sequeira et al., 1991; Baveco and Smeulders, 1994; Holst et al., 1997; Hill et al., 1998; Alfredsen and Sæther, 2000; Spanou and Daoyi, 2000). The benefits of using an object-oriented approach for ecological modeling and simulation are now well recognised (Silvert, 1993). Components extend object-oriented concepts to create a framework for dealing with hierarchical (sub-)systems interacting through well defined interfaces. In addition, ecological models take advantage of component-based design in several ways (Meyer, 1988 in Papajorgji et al., 2004): to improve reusability and ease maintenance, to enable cross-platform computing, and to combine distributed components through a service-oriented architecture. In this paper, we consider *components* to be self-contained systems with well-defined interfaces that send and receive discrete events.

The principle of activity tracking follows naturally from this view of components as discrete-event systems (Muzy and Zeigler, 2008). Simulation algorithms that are based on activity tracking exploit the natural, dynamic variation of a process in time and space. Although complexity of environmental models can be reduced *a posteriori* (Lawrie and Hearne, 2007), *a priori* design, analysis and implementation structures are provided here to build efficient computational models. These algorithms are both efficient and easy to apply when the model is conceived of as a collection of components that interact through discrete events. Activity focused modeling, using the constructs of the DEVS formalism and event driven simulations, is a coherent approach to building efficient simulation models of dynamic, spatially distributed systems. This coherent world-view is particular useful for navigating the numerous methods that have been proposed for building models of complex ecological systems (Sequeira et al., 1991; Silvert, 1993; Baveco and Smeulders, 1994; Holst et al., 1997; Hill et al., 1998; Alfredsen and Sæther, 2000; Spanou and Daoyi, 2000).

This paper is organized as follows. In Section 2, a brief review of the state-of-the-art in forest fire modeling and simulation is provided, and the activity tracking paradigm is introduced and applied to a physics-based model. In Section 3, the activity-based simulations presented. In Section 4, the whole approach is discussed. Section 5 summarizes and concludes the paper.

2. Material and methods

Forest fires have well known economic, ecological, health and social impacts. The huge forest fires in Greece, Portugal, Australia and California that occurred in 2007 are recent reminders of these costs, and the acres of forest burned by wild fires each year is increasing exponentially (*cf.* Fig. 1).

Modeling fire spread behavior is a hard task. In physics, very precise models focusing on relevant generic mechanisms (convection, diffusion and radiation) are usually experimented with by scientists to improve their knowledge. To develop generic, easy to modify, and efficient implementations of these models is very challenging.

2.1. *Forest fire modeling*

The behavior of forest fires is better understood now than it was 20 years ago. Significant research has been conducted in connection with the forest fires, particularly on the conditions favorable to ignition (period and intensity of hydric deficit: Dolling et al., 2005; Núñez-Regueira et al., 2001, type of fuel: Dong Hyun et al., 2006), thermochemical and physicochemal characteristics, biological traits, bioclimatic parameters of various type of vegetation (Núñez-Regueira et al., 2000), and on the conditions of their propagation (topography, direction and intensity of the winds, connectivity of the surfaces covered by vegetation; Thompson et al., 2000).

Forest fire models can be separated into stochastic and deterministic (analytical) models. Stochastic models aim at predicting the most probable fire behavior in average conditions, using evolution rules drawn from lab experiments and field data samples. On the contrary, in analytical models, the fire behavior is usually deduced from the physical laws driving the evolution of the system. Recently, several sophisticated models were proposed (Barros and Mendes, 1997; Karafyllidis and Thanailakis, 1997; Hernández Encinas et al., 2007; Yassemia et al., 2008) and successfully validated by comparison with real fires. All of these models use either simple cellular automata or dynamical structure cellular automata (DSCA).

Based on Weber's classification (Weber, 1990), three kinds of mathematical models for fire propagation can be identified according to the methods used in their construction. The first type of models are *statistical* models (McArthur, 1966), which make no attempt to include specific physical mechanisms, being only a statistical description of test fires. The results can be very successful in predicting the outcome of fires similar to the test fires. However, the lack of a physical basis means that the statistical models must be used cautiously outside the test conditions. The second category of models is *semi-empirical* models (Rothermel, 1972) based on the principle of energy conservation but which do not distinguish between the different mechanisms of heat transfer. Rothermel's stationary model is a one-dimensional model, in which a second dimension can be obtained using propagation algorithms (Richards, 1990) integrating wind and slope. Finally, *physical* models (Albini, 1985) integrate wind and slope effects in a more robust manner by describing the various mechanisms of heat transfer and production. Physical mechanisms are described using a

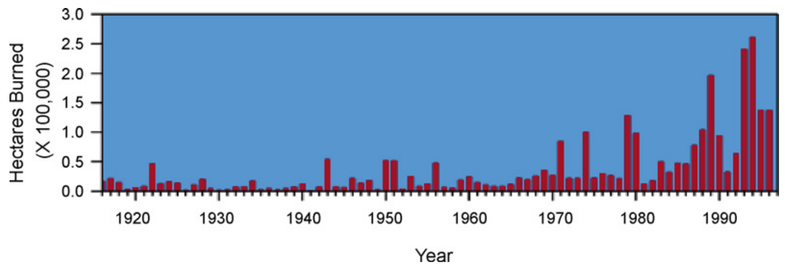


Fig. 1 – Annual area burned by wildfires in Arizona and New Mexico (http://geochange.er.usgs.gov/sw/impacts/biology/fires_SOI/).

chemical, thermal and mechanical definition of basic fire phenomena. Hence, physical and semi-empirical models use the definition of basic fire phenomena to physically describe fire propagation.

Today, most ready-to-run software for fire spread simulation (Veach et al., 1994; Finney, 1995; Coleman and Sullivan, 1996; Albright and Meisner, 1999; Lopes et al., 2002) and simulations of fire spreading on large-scale (Wu et al., 1996; Hargove et al., 2000; Miller and Yool, 2002) are based on Rothermel's model (Rothermel, 1972). A lot of effort has been placed in improving simulations of Rothermel's model. In the CA field, studies pinpoint the need for developing new classes of CA for fire spreading applications (Karafyllidis and Thanailakis, 1997; Berjak and Hearne, 2002). Many methods based on discrete-event formalisms and object-oriented programming have been proposed to improve CA capabilities for fire spread simulation (Vasconcelos et al., 1995; Ntaimo and Zeigler, 2005; Ameghino et al., 2001; Muzy et al., 2003). Unlike basic CA, these models can receive external update information, the fire perimeter can be updated at any moment due to the continuous time nature of the discrete-event specifications, and active cells can be dynamically created and removed to save memory for large cell spaces.

2.2. *Activity tracking framework*

The appropriate choice of a time management scheme depends on the nature of the system and on the modeling objectives. For a system in which every state change occurs at a fixed increment of time, discrete events will produce simulation overhead, and a discrete-time driven simulation will be more efficient because we do not have to predict (by computation) what we already know will happen and when. However, in natural systems discrete-time evolution does not exist. Discrete-time flows in a model exist only after the discretization of continuous time by the modeler (Ralston and Rabinowitz, 2001). For other systems, "while other formalisms allow representation of space and resources, only discrete-event models offer the traditional ability to explicitly and flexibly express time and its essential constraints on complex adaptive systems behavior and structure" (Zeigler, 2005). The advantage of discrete-event driven simulations is that a simulation model evolves directly from one state change to another. No computations are performed during inactive periods, but this requires that the next state change can be forecast from the current state and it requires an efficient method for scheduling events.

Fig. 2 depicts the essential parts of an activity tracking simulation. It merges the three usual world-views (activity-, event-, and process-oriented strategies) into a single framework that includes discrete-time driven models as well; the usual world-views are underlined where they appear in the activity tracking strategy. Marks are added to and removed from components to track activity as it propagates through the model; these marks are used to maintain the set of active components as the simulation progresses. Every activity tracking simulation has two basic steps:

- *Step I*: Components exchange information and the propagation of activity is tracked. The current active set is scanned for components that have output events and these events are routed to their destinations; routing in hierarchal models can be done recursively (for more information, see Muzy and Nutaro, 2005). The active set is updated to include atomic components that have received these output events. Atomic components are basic behavioral components that are coupled together to build a hierarchical structure (for more information, see Zeigler et al., 2000).
- *Step II*: New states are computed for active components from their current states and inputs. Components that changed state significantly are marked and added to the active set; components that do not undergo a significant change of

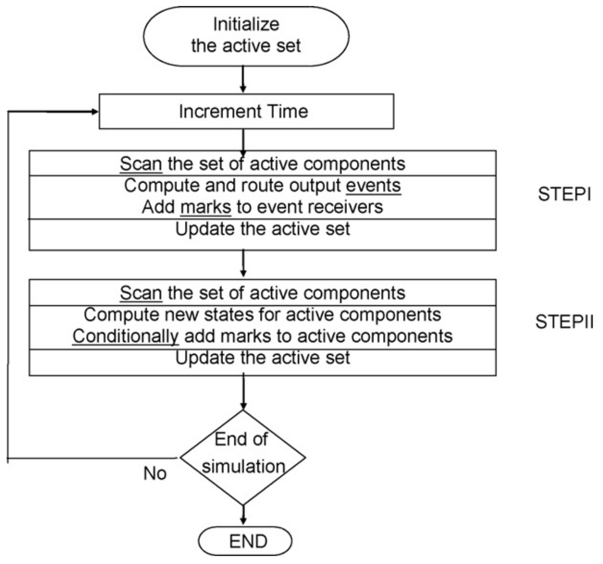

Initialize the active set
Increment Time
Scan the set of active components
Compute and route output events
Add marks to event receivers
Update the active set
STEPI
Scan the set of active components
Compute new states for active components
Conditionally add marks to active components
Update the active set
STEPII
End of simulation
No
END

Fig. 2 – State-centric activity tracking pattern.

state are removed from the active set. In a discrete-event driven simulation, the active set is an event scheduler and the active components are in the schedule or will receive an input from a model in the schedule.

This a state-centric two-step pattern. Both local and global levels of modularity for state transitions can be achieved. Local transitions are achieved on states of single components. Global transitions are achieved on many component states. More details can be found in Muzy and Zeigler (2008).

2.2.1. *Dynamic structure cellular automata (DSCA) specification*

Cellular models are commonly used to model ecological systems because spatial relationships are a crucial to how ecological systems function (Wu and David, 2002). Regardless of the simulation kernel used, be it time driven or event driven, the simulation algorithm rarely focuses computations exclusively on active cells, but carefully scrutinizes every cell in the grid. This type of implementation is conceptually different from the activity focused evolution of the real process and at odds with the final analysis where we focus on change, not its absence. More practically, this approach to simulation requires tremendous computational resource and consequently is limited with regards to the physical space it can handle. For instance, a forest growth simulator developed by some of us cannot reasonably manage more than 25*Ha* within a simulation over one century on a fast running desktop computer.

The ability of cellular automata to change topology has been investigated through two research directions. First, structurally dynamic cellular automata (SDCA) (Alonso-Sanz, 2007) study a phenomenology of neighborhood changes during the simulation. Discrete-value cells change neighborhood according to their own value. Second, DSCA (Barros and Mendes, 1997; Muzy et al., 2004) are based on the System Theory. Mathematical structure specifications and simulation algorithms are provided which constitute a generic framework for implementing simulation models. DSCA models specify structure as a function of the model's complete state, and the structure of the model can change as the states of its components change. DSCA can be specified as discrete-time or discrete-event networks.

We present here a network specification of dynamic structure cellular automata (DSCA, *cf.* Fig. 3):

$$\text{Network}_{\text{DSCA}} = \langle X_{\text{DSCA}}, Y_{\text{DSCA}}, M_\chi \rangle$$

where X_{DSCA} and Y_{DSCA}, are respectively the input and output sets.

Dynamic structure changes are handed by the executive model:

$$M_\chi = \langle X_\chi, Y_\chi, S_\chi, \delta_\chi, \lambda_\chi, \tau_\chi \rangle$$

The *structural state* is defined as $S_\chi = \langle D, \{C_i\}, \{I_i\}, \{Z_{i,j}\} \rangle$, which can be described as follows:

- D contains the references i (coordinates or number) of *active* cells,
- $\{Z_{i,j}\}$ is the set of coupling functions. Coupling functions describe how components are connected:

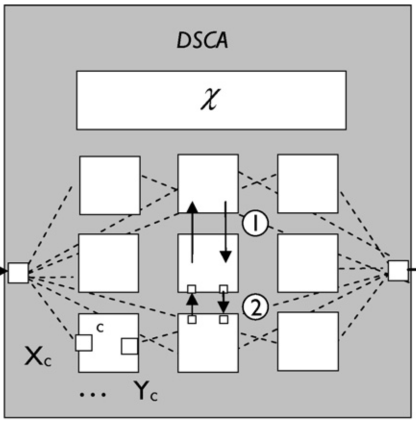


Fig. 3 – A dynamic structure cellular automata. The DSCA component has input and output sets X_{DSCA} and Y_{DSCA} constituted of ports (represented as boxes) to interact with external components in a modular way. A modular specification allows the encapsulation and increasing behavioral autonomy of components. However, interacting massively through discrete events and ports has a simulation cost. Case (1) corresponds to a non-modular specification of cells (transition functions of cells directly influencing each others). Case (2) corresponds to a modular specification of cells. The network executive χ is specified here in a non-modular way (for efficient access of component states). Coupling between cells and DSCA inputs and outputs are indicated as dash lines. When the non-modular specification is chosen, the corresponding coupling function does not need to be specified.

  - ○ The network executive χ is embedded in the DSCA: $Z_{\text{DSCA}\to\chi}$: $X_{\text{DSCA}} \to X_\chi$ and $Z_{\chi\to\text{DSCA}}$: $Y_\chi \to Y_{\text{DSCA}}$;
  - ○ The network executive χ can be connected to a cell $c \in D$: $Z_{\chi\to\text{cell}}$: $Y_\chi \to X_c$ and $Z_{c\to\chi}$: $Y_c \to X_\chi$;
  - ○ A cell can be externally connected to both input X_{DSCA} and output Y_{DSCA} of the DSCA: $Z_{\text{DSCA}\to c}$: $X_{\text{DSCA}} \to X_c$, $Z_{c\to\text{DSCA}}$: $Y_c \to Y_{\text{DSCA}}$;
  - ○ A cell can be internally connected to its neighboring cell $p \in D$ as: $Z_{p\to c}$: $Y_p \to X_c$ and $Z_{c\to p}$: $Y_c \to X_p$;
- For a cell $c \in D$, $I_c = \{N_c, \text{DSCA}, \chi\}$, $I_\chi = \{\{C_i\}\}$ where N_c is the neighborhood of a cell c.

  δ_χ:$X_\chi \times S_\chi \to S_\chi$, is the structural state transition function. According to current structural state and inputs, the transition function computes a new structural state. Changes in structure include changes in cells neighborhoods, changes in cell definitions, and addition or deletion of cells. The structural state transition function is composed of internal and external functions $\delta_\chi = \left\{\delta_{\text{int}_\chi} \cup \delta_{\text{ext}_\chi}\right\}$. External transitions account for external events and internal transitions for autonomous computations (for more information: Barros, 1997).

  λ_χ: $S_\chi \to Y_\chi$ is the structural state output function. Through the output function structural states can be sent to other models.

  $\tau_\chi : S_\chi \to \mathbb{N}$ for a discrete-time base and $\tau_\chi : S_\chi \to \mathbb{R}^+$ for a discrete-event time base.

As a minimum assumption, each cell c can be specified as an atomic component:

$C_i = \langle X_i, Y_i, S_i, \delta_i, \lambda_i, \tau_i \rangle$

where

X_i and Y_i, are respectively input and output sets.
$S_i = \langle (m, n), S^{N_i}, \text{phase} \rangle$, with

$$\begin{cases} S^{N_i} = \{ s_p / p \in I_i \} \\ \text{phase} = \{\text{“active”}, \text{“passive”}, \ldots\} \end{cases}$$

where $(m, n) \in \mathbb{N}^2$ are cells' coordinates, N_i has been defined previously as the neighborhood set, S^{N_i} represents the states of neighboring cells. When receiving or sending its state, a cell is in phase “*active*”, otherwise it is in phase “*passive*”.
δ_i: $X_i \times S_i \to S_i$ is the transition function[1] composed of internal and external functions $\delta_i = \{\delta_{\text{int}_i} \cup \delta_{\text{ext}_i}\}$, where $\delta_{\text{int}_i} : S_i \to S_i$, and $\delta_{\text{ext}_i} : X_i \times S_i \to S_i$.
λ_i: $S_i \to Y_i$ is the output function.
$\tau_i : S_i \to \mathbb{N}$ for a discrete-time base and $\tau_i : S_i \to \mathbb{R}^+$ for a discrete-event time base.

For a more complex cell, the latter can be decomposed as a network (dynamic structure or not) of sub-components (cf. sub-section 2.3.6). However, regarding the closure under coupling of dynamic structure networks, precise network specifications can be expressed by (or is equivalent to) a single atomic specification (more details in Zeigler et al., 2000).

2.3. *Fire spread application*

Models derived from basic physical processes describe the spread of fire in terms of well-understood heat propagation principles (radiation, convection, etc.) These continuous mathematical models can be discretized in several different ways for the purposes of simulation using discrete event and discrete-time methods. Our work extends the model presented in (Muzy et al., 2005a,b) to include an implicit and a quantization-based descretization through activity-based modeling and simulation.

2.3.1. *The physics-based model*

Physics-based fire spread models usually consist of reaction diffusion equations. These PDEs cannot be solved analytically. They have to be solved numerically. Discrete-time solutions can be obtained using Explicit (Forward) or Implicit (Backward) Euler methods.

The physical model we use (Balbi et al., 1998) is composed of elementary cells of earth and plant matter. Under no wind and no slope conditions, the temperature of every cell is represented by the following PDE:

$$\frac{\partial T}{\partial t} = -k(T - T_a) + K\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right) - Q\frac{\partial \sigma_v}{\partial t} \tag{1a}$$

$$\sigma_v = \sigma_{v0} \quad \text{if } T < T_{ig} \tag{1b}$$

$$\sigma_v = \sigma_{v0} \cdot e^{-\alpha(t - t_{ig})} \quad \text{if } T \geq T_{ig} \tag{1c}$$

$$T(x, y, t) = T_a \quad \text{at the boundary} \tag{1d}$$

$$T(x, y, t) \geq T_{ig} \quad \text{for the burning cells} \tag{1e}$$

$$T(x, y, 0) = T_a \quad \text{for the non-burning cells at } t = 0 \tag{1f}$$

where T_a (27 °C) is the ambient temperature, T_{ig} (300 °C) is the ignition temperature, t_{ig} (s) is the ignition time, T (°C) is the temperature, K (m^2 s^{-1}) is the thermal diffusivity, Q (m^2 °C/kg) is the reduced combustion enthalpy, α (s^{-1}) combustion time constant, σ_v (kg m^{-2}) is the vegetable surface mass, and σ_{v0} (kg m^{-2}) is the initial vegetable surface mass (before the cell's combustion). The term $k(T - T_a)$ represents thermal exchanges with the air, the term $K((\partial^2 T/\partial x^2) + (\partial^2 T/\partial y^2))$ represents the diffusion phenomenon, and the term $Q(\partial \sigma_v/\partial t)$ represents the combustion energy (the reaction).

2.3.2. *Discrete-time solutions*

In a previous study, Finite Element and the Finite Difference Methods were used to discretize this physical model (Santoni, 1996). The Finite Difference Method provided equivalent results as the Finite Element Method. However, the latter appeared more complex to implement and involved longer execution times. Therefore, the Finite Difference Method has been chosen.

2.3.3. *Explicit solution*

The physical model (1a)–(1f) solved by the Explicit Method leads to the following difference equation:

$$T_{i,j}^{n+1} = a(T_{i-1,j}^n + T_{i+1,j}^n) + b(T_{i,j-1}^n + T_{i,j+1}^n) + cQ\left(\frac{\partial \sigma_v}{\partial t}\right)_{i,j}^n + dT_{i,j}^n + e \tag{2}$$

where T_{ij} is the grid node temperature. The coefficients a, b, c, d and e depend on the time step and the mesh size considered.

The study domain is meshed uniformly with cells of 1-cm^2 and a time step of 0.01 s. This is a very small time and space scale. However, this small scale allows us to be consistent in terms of both error and physical descriptions of a fire spread thus achieving a precise analysis of simulation results.

The propagation domain consists of a cellular model where each future cell temperature is calculated using the current cell temperature and temperatures of the cardinal neighbors.

2.3.4. *The implicit solution*

Implicit methods are typically more stable than explicit methods and allow larger time steps thus reducing execution times. The physical model (1a–f) solved by the Implicit Method leads to the following difference equation:

$$\left| T_{i,j}^{k+1,n+1} - T_{i,j}^{k,n+1} \right| < \varepsilon \tag{3a}$$

$$T_{i,j}^{n+1} = a'(T_{i-1,j}^{n+1} + T_{i+1,j}^{n+1}) + b'(T_{i,j-1}^{n+1} + T_{i,j+1}^{n+1}) + c'Q\left(\frac{\partial \sigma_v}{\partial t}\right)_{i,j}^n + d'T_{i,j}^n + e' \tag{3b}$$

[1] Modularity cases: (1) $S_i = S_i^{N_i}$ and $X_i = X_{DSCA}$ (assuming external influences of cells), (2) $X_i = X_{DSCA} \times X_i^{N_i}$, with $X_i^{N_i} = \{x_p / p \in I_i\}$.

```
$\delta_\chi$ ($s_\chi$, e, $x_\chi$) :
 For each cell $i \in D$ Do
  If($s_{ci}$.phase == "passive") Then
     remove $c_i$ from D
  Endif

  If($s_{ci}$.phase == "testing") Then
   If (ci.m == K OR ci.m == 0 OR ci.n == 0 OR ci.n == L) Then
    $s_{ci}$.phase == "non_testing"
   Else
    If( $\left|s_{ci}.T_{i,j}^{k+1} - s_{ci}.T_{i,j}^{k}\right| > \theta$ )Then
     $s_{ci}$.phase == "non_testing"
     add neighboring cells $c_p$ to D, with $p \in N_{ci}$
    EndIf
   EndIf
  EndIf

 EndFor
```

Fig. 4 – Transition function of the DSCA.

where k is the iteration step. The coefficients a', b', c', d' and e' depend on the time step and on the mesh size considered.

The solution is calculated at each time step (of 0.1 s) using the iterative method of Jacobi (Sibony and Mardon, 1988) for which the convergence condition (3a) is used to pass on to the next time step. As long as all the whole temperatures calculated between two successive iterations are not less than an ε (10^{-3} K for instance), the simulation clock is not incremented.

In this instance, the problem could be solved with a Low Up decomposition. However, to end up with a CA structure, the Jacobi method is used. Hence, performance metrics can be compared with other explicit and quantization-based CA structures.

2.3.5. *Dynamic structure cellular automata solution*

To focus the simulation on active cells, we use the basic principles exposed in (Zeigler et al., 2000) to predict whether a cell will possibly change state or will definitely be left unchanged in a next global state transition: "a cell will not change state if none of its neighboring cells changed state at the current state transition time". Nevertheless, to obtain optimum performance the entire set of cells cannot be tested. Thereby, an algorithm, which tests only the neighborhood of the active bordering cells of a propagation domain, has been defined for this type of phenomena. We specify here a DSCA as a dynamic structure discrete-time network embedding initial ignitions. This DSCA is used here only to track active cells (adding and removing them from the active set).

As depicted in Fig. 4, the activity tracking algorithm is located in the global transition function (δ_χ) of the DSCA, in charge of the structure evolution of the cell space. *ci.m* and *ci.n* correspond to the coordinates (m, n) of a cell c of reference $i \in D$ (also abstractly named as ci). *Sci.phase* corresponds to the phase of state $s \in S$ of a cell i. An activity state ('*inactive*', '*active*' and '*testing*') is added to the cells. The activity state '*testing*' is used to track new activated components. Cells are considered in '*testing*' phases when located at the edge of the propagation domain, '*non_testing*' when not tested at each state transition, and '*passive*' when inactive. Remember that the specification of cell structures (neighborhood, transition functions, etc.) is statically pre-allocated.

A propagation example is sketched in Fig. 5 for cardinal and adjacent neighborhoods. In our algorithm, only cells at the leading edge of the fire test their neighborhood for cells that should become active. This minimizes the number of cells that must be tested for activation at each iteration of the simulation algorithm.

The result of the executive transition function of Fig. 4 depends on the state of the tested cells. Changes of cells states are checked between two time steps ($|s_{ci} \cdot T_{i,j}^{k+1} - s_{ci} \cdot T_{i,j}^{k}|$), are compared to a threshold θ defined by the user. If $|s_{ci} \cdot T_{i,j}^{k+1} - s_{ci} \cdot T_{i,j}^{k}| > \theta$, then the cell's neighbors become active and are marked as '*testing*', the cell itself is marked as '*non_testing*'. Notice here the difference between a *quantum* and a *threshold*. A quantum consists of discretizing a state in equal finite values. Considering a particular cell, during the whole simulation, times of boundary crossings are computed continuously. Conversely, a threshold is used to determine if a cell turns active or not. If a cell's state is greater than a fixed threshold, the cell turns '*non_testing*' and the threshold test is not used anymore, for that particular cell, during the simulation.

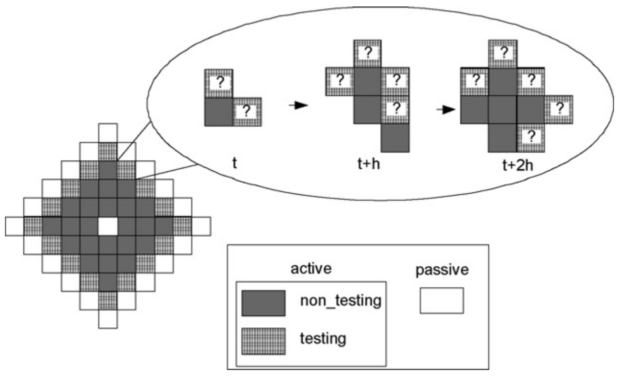


Fig. 5 – Calculation domain evolution.

2.3.6. *Quantization solution*

Quantization of ODEs has been introduced in (Zeigler et al., 2000; Bolduc and Vangheluwe, 2002; Nutaro, 2003). Instead of generating approximate solutions at discrete points in time, the solution is approximated by looking at significant changes in the system's state. Efficiency and stability of this discretization technique are discussed in (Nutaro and Zeigler, 2007). The magnitude of change in the solution that is considered to be significant is called an integration quantum (or quantum). A quantum can be defined by

$$D = |\Phi^{n+1} - \Phi^{n}| \tag{4}$$

where D is the desired change in the solution at each step of the computation and Φ^{n+1} and Φ^{n} are two numerical approximations of the continuous function $\Phi(t)$ representing a time invariant process.

To illustrate the basic elements of quantized integration methods, a quantized integration scheme for solving a single ODE can be constructed using the explicit Euler formulae

$$\Phi^{n+1} = \Phi^{n} + \Delta t \cdot f(\Phi^{n}). \tag{5}$$

The time required for a quantum change to occur is approximated by

$$\Delta t = \frac{D}{|f(\Phi^{n})|} \tag{6}$$

If $f(\Phi^{n}) = 0$, then $\Delta t \to \infty$, which indicates that an equilibrium state has been reached.

The quantized integration scheme is constructed by substituting Eq. (6) into Eq. (5) and keeping track of the sign of the derivative to ensure that the solution moves in the proper direction. This gives the system

$$\Phi^{n+1} = \Phi^{n} + D \cdot \mathrm{sign}(f(\Phi^{n})) \tag{7}$$

which approximates successive values of the continuous system. The time $t_{n+1} \in \Re$ of approximated states is given by

$$t_{n+1} = t_n + \frac{d}{|f(\Phi^{n})|} \tag{8}$$

An approximation example of the function $\Phi(t)$ is sketched in Fig. 6. One can notice that for a continuous time base, the number of computations needed to approximate the curve $\Phi(t)$ can be reduced, even more when the latter changes slowly during time.

Eqs. (7) and (8) constitute a quantized integrator. In space, and when applied to PDEs, using such integrators allows to track activity, that is, changes in time in one point of the space. Then, each integrator can be implemented as an atomic model DEVS.

Quantization of Eq. (1a) consists of discretizing the term of thermal exchanges, the diffusion term and the reaction term. Then, in every cell a quantized integrator [*cf.* Eqs. (7) and (8)] calculates the temperature of the cell according to the value of: (1) the term of thermal exchanges (which is discretized), (2) the term of diffusion (which is discretized), and (3) the reaction term (which is quantized).

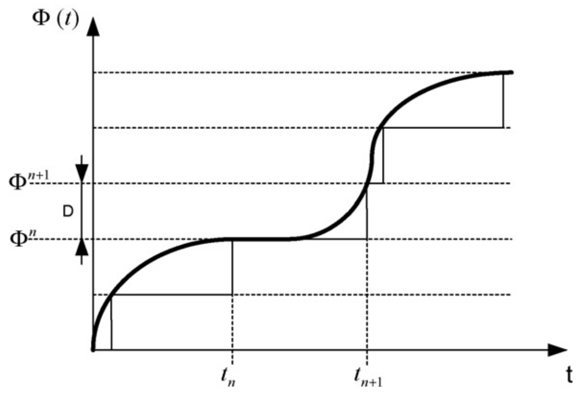


Fig. 6 – Quantization approximation of ODE.

The discretization of the diffusion term in space using center differences leads to:

$$K\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right) \approx K\left(\frac{T^{n}_{i,j-1} - 2T^{n}_{i,j} + T^{n}_{i,j+1}}{(\Delta x)^2} + \frac{T^{n}_{i,j-1} - 2T^{n}_{i,j} + T^{n}_{i,j+1}}{(\Delta y)^2}\right) \tag{9}$$

The term of thermal exchanges can be directly discretized:

$$K(T - T_a) = K(T^{n}_{i,j} - T_a) \tag{10}$$

When the cell is burning, the energy $Q(\partial\sigma_v/\partial t)$ released by the combustion depends directly on the exponential decrease of the fuel mass:

$$\sigma^{\nu} = Q\sigma^{\nu^0}\ e^{-\alpha(t-t_{ig})}, \quad \text{if } T \geq T_{ig} \tag{11}$$

By knowing that the fuel mass will decrease only by one quantum D, we obtain:

$$\sigma_\nu - D = Q\sigma_{\nu 0}\ e^{-\alpha t_a} \tag{12}$$

Taking the logarithm, we obtain the time advance t_a:

$$t_a = \frac{1}{\alpha}\ \ln\frac{\sigma_\nu - D}{Q\sigma_{\nu 0}} \tag{13}$$

Every cell is a coupled model composed of two atomic models: one model implements Eqs. (12) and (13) the other solves Eqs. (9) and (10). Fig. 7 describes this coupled model (CM) which is composed of an atomic model of Fuel Decrease (AM_{FD}) and of an atomic model of Thermal Exchanges and Diffusion (AM_{ED}).

An atomic model AM_{ED} is connected to its four cardinal neighbors through eight ports. Cells' temperatures are sent and received through these ports. An atomic model AM_{FD} describes the evolution of the fuel mass decrease when the cell is burning. After the reception of the cell's temperature (contained in AM_{ED}), if the cell is in combustion, the external transition function of AM_{FD} will compute the time of next quantum boundary crossing according to the time advance (13). At each quantum crossing, computed by an internal tran-

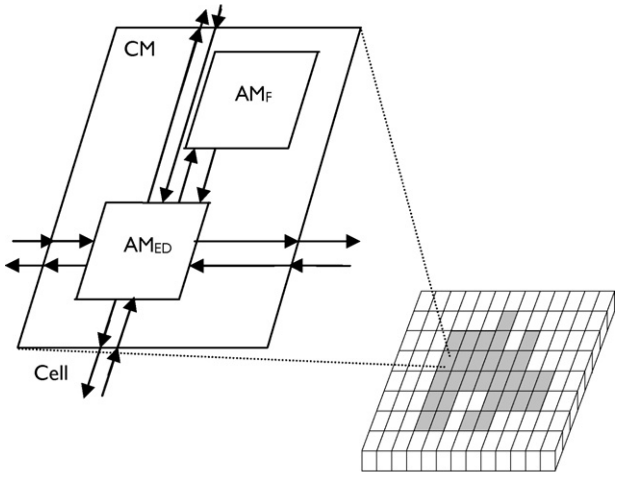


Fig. 7 – Cell description.

sition, influenced cells are updated immediately. Finally, the derivative of the temperature, calculated by AM_{ED}, according to the neighboring cells' temperatures and to the fuel mass, is integrated using Eqs. (7) and (8).

3. Simulation results

To validate the simulation results, experimental fires have been conducted on *Pinus Pinaster* litter, in a closed room without any air motion, at the INRA (Institut National de la Recherche Agronomique) laboratory near Avignon, France (Balbi et al., 1998). These experiments have been performed in order to observe fire spread for point-ignition fires under no slope and no wind conditions. The experimental apparatus was composed of a one square meter aluminum plate protected by sand. A porous fuel bed was used, made up of pure oven dried pine needles spread as evenly as possible on the total area of the combustion table (in order to obtain a homogeneous structure). The experiment consisted in igniting a point using alcohol. The resulting spread of the flame across the needles has been closely observed with a camera and thermocouples.

3.1. *Particularity and validation of the different methods*

Fig. 8 depicts a visual comparison between the simulated fire fronts (using explicit, implicit and quantization methods) and experimental fire fronts (represented by dots). The experimental fire front corresponds to pictures of actual fire spread positions under laboratory experiments on pine needle substrates. Explicit, implicit and quantization solutions allow simulating precisely the fire front positions. The quantized simulation focuses its computational effort on the active cells. The explicit and implicit CA do not; they do the same amount of work to simulate each cell, regardless of its rate of change.

Fig. 9 depicts the various evolutions of the simulated fire spread for many quantum sizes. For large quantum sizes, a qualitative simulation can be achieved; although the propagation is delayed, the circular shape of the fire front is conserved. Besides, for an actual propagation of 221 s, an execution time of 10 s is obtained on a 1.5 GHz *Pentium M Centrino* (this computer is used for all experiments described in this paper). Qualitative simulations can be used to reduce the testing phase of model development (by execution time reductions), as well as to improve the rapidity of predictive fire spread simulations. These simulations are usually used by fire fighter command centers to predict *where* will spread the fire. They do not care *when*, the only thing they have to know is *where* the fire will spread to set fire fighters and vehicles.

Fig. 10 depicts the focus of both quantized and DSCA simulations on active cells. The fire fronts are represented by lines. Both simulations adaptively track the fire front evolutions by changing the set of active cells as the simulation is executed. In Fig. 10, the result of both methods is equivalent for focusing on active cells.

Fig. 11 depicts how the quantized simulation focuses on fire fronts for different thresholds. One can notice that few errors are introduced by truncating information, even when the threshold is relatively large (up to 30 K). However, for a quantum of 40 K (and higher), the fire front becomes noticeably distorted.

3.2. *Quantization method*

The fuel mass decrease simulated by the atomic model AM_{FD} is represented in Fig. 12. The end of this negative exponential curve pinpoints the interest of using a discrete-event simulation to pass directly between significant (as defined by the integration quanta) values of the continuous state. This allows reducing the number of state transitions. However, the quantum size of this atomic model has to be carefully chosen because of the important beginning decreasing slope of the exponential curve. A too big quantum size leads to significant errors. A quantum of 0.01 kg/m has been chosen for this model.

As Eq. (1a) does not have analytical solution, the quantization simulation results are visually validated through the laboratory experiment. Moreover, an average relative error is calculated against the explicit simulation results already validated through numerous studies (Balbi et al., 1998):

$$\bar{\varepsilon} = \sum_{i=1}^{N} \frac{1-(q_i/q_i^*)}{N}$$

With N = number of cells, q_i = explicit value of cell i and q_i^* = numerical value of cell i.

Fig. 13 depicts the average relative error for different quantum sizes of the AM_{ED} atomic model, and the same experiment. This curve is almost linear. For very small quantum values, the error is constant. This means that the quantization solution is very close to the actual propagation. Fig. 13 depicts too the number of transitions (which is proportional to the execution time) of the quantization solution, according to different quantum sizes. Increasing the quantum size reduces the number of transitions. Moreover, until a quantum size of 5 K, the number of transitions decrease rapidly. For the quantization solution, a quantum size of 5 K is chosen in the rest of the paper, leading to an approximate error of 5%.

Fig. 14 compares execution times of the explicit, implicit, and quantization solutions, for different simulated domain

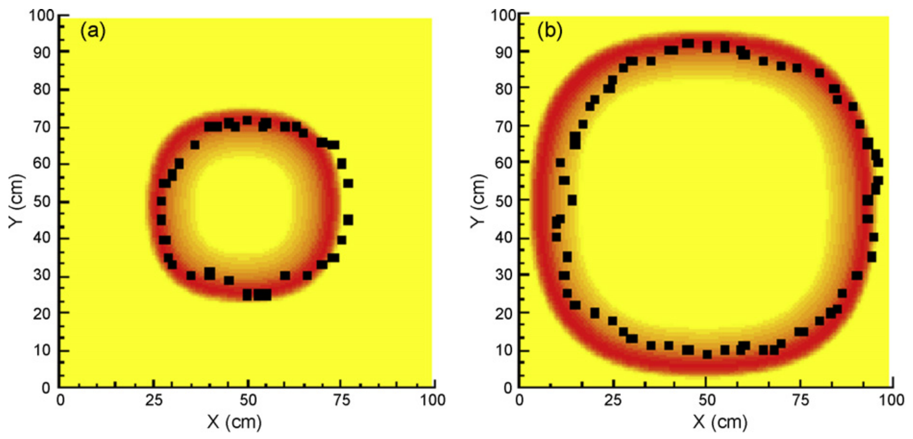


Fig. 8 – Comparisons at (a) 75 s and (b) 122 s between simulated (gray scale) and experimental front fires (dotted line). Both simulations have been obtained by quantization model ($D = 1$ K), implicit CA ($\Delta t = 0.1$ s) and explicit CA ($\Delta t = 0.01$ s).

sizes that result from using different numbers of cells (changing the resolution of the spatial discretization). Concerning the implicit solution, using a bigger time step (remember that we have a time step of 0.1 s for the implicit method and 0.01 s for the explicit one) allows to reduce execution times. However, quantization execution times are significantly smaller.

The good results obtained for execution times can be explained by looking at Fig. 15, which depicts the number of transitions of the different methods for different propagation domain sizes. Indeed, the quantization method greatly reduces the number of transitions.

3.3. *Dynamic structure cellular automata*

In the DSCA method, different quantum sizes can be chosen for both implicit and explicit methods. Errors, number of transitions and execution times of DSCA (implicit and explicit) are compared to CA (implicit and explicit).

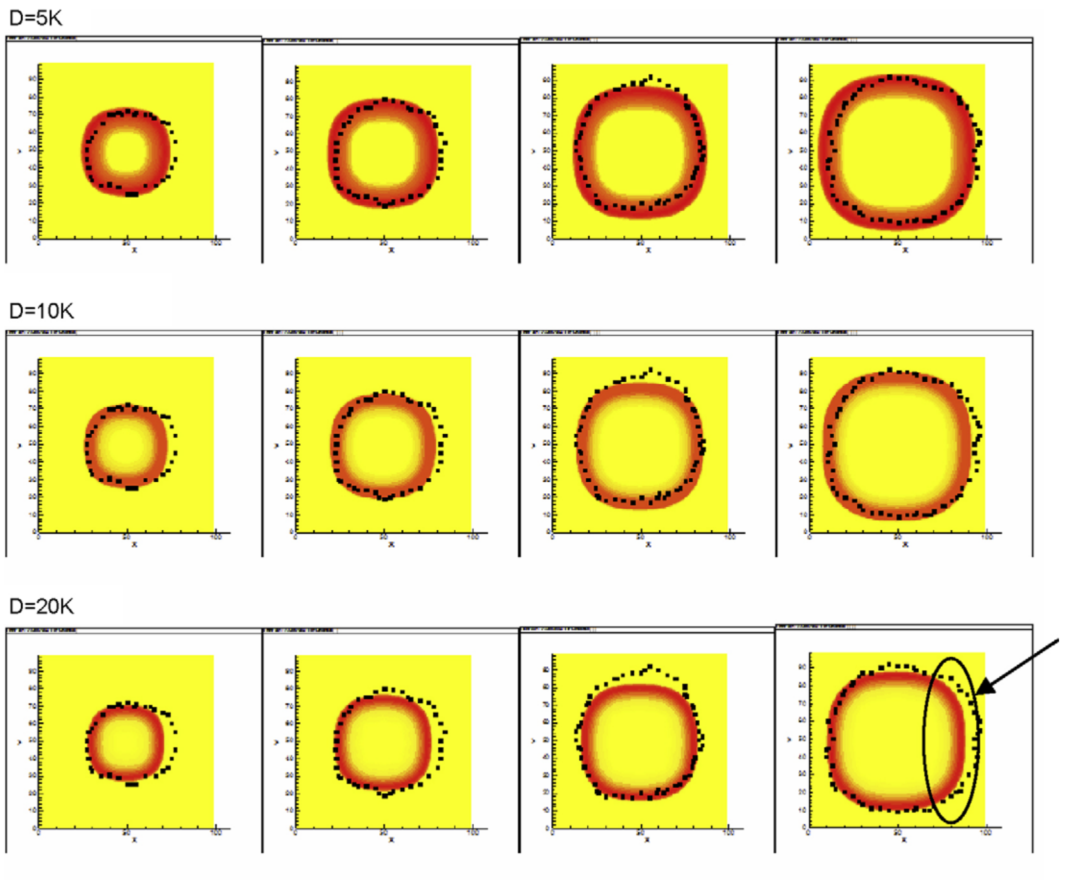


Fig. 9 – Comparison for various quantum sizes *D* between simulated (gray scale) and experimental front fires (dotted line).

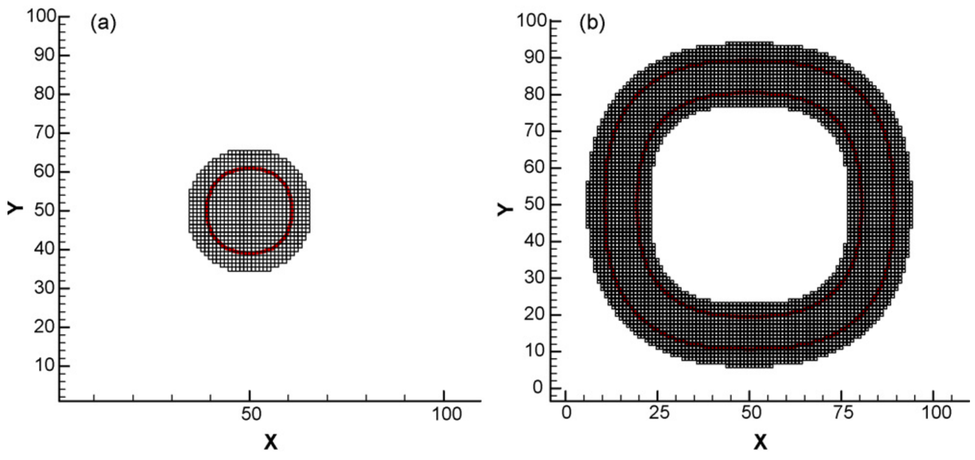


Fig. 10 – Focus of the quantization and DSCA simulations on active cells at: (a) 75 s and (b) 122 s (quantization method: $D = 1$ K, implicit DSCA: $\Delta t = 0.1$ s and $\theta = 1$ K, and explicit DSCA: $\Delta t = 0.01$ s and $\theta = 1$ K).

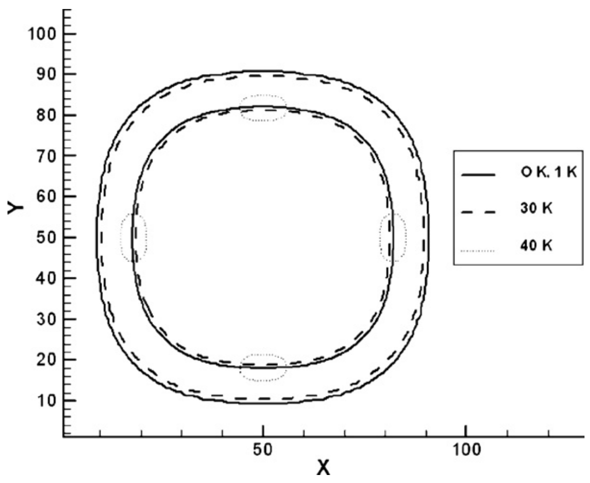


Fig. 11 – The DSCA explicit focus (explicit DSCA: $\Delta t = 0.1$ s).

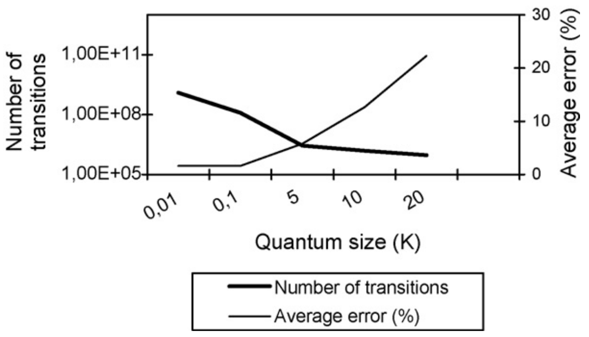


Fig. 13 – Number of transitions and error for different quantum sizes (quantization method).

3.3.1. *Implicit method in dynamic structure cellular automata*

Fig. 16 depicts the average relative error obtained for different threshold sizes. This error is linear and tends to zero for small threshold sizes, thus exhibiting the precision of the method for small thresholds. Fig. 16 depicts too the number of transitions (which is proportional to the execution time) for different threshold sizes when simulating DSCA implicit schemes. We can notice that reducing the quantum size, the reduction in the number of transitions, and consequently in the execution time, is not as great as for the quantization method.

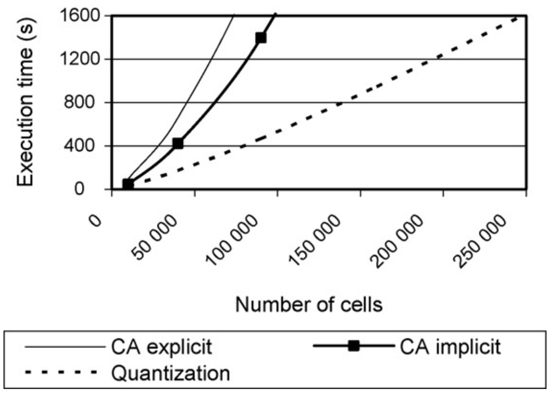


Fig. 14 – Execution times for different domain sizes (quantization model: $D = 5$ K, implicit CA: $\Delta t = 0.1$ s, explicit CA: $\Delta t = 0.01$ s).

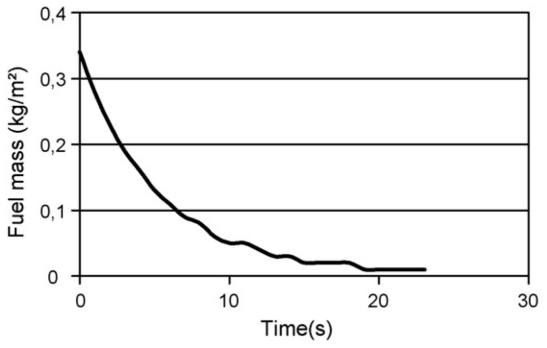


Fig. 12 – Fuel mass decrease using a discrete-event time base (quantization method, $D = 0.01$ kg/m).

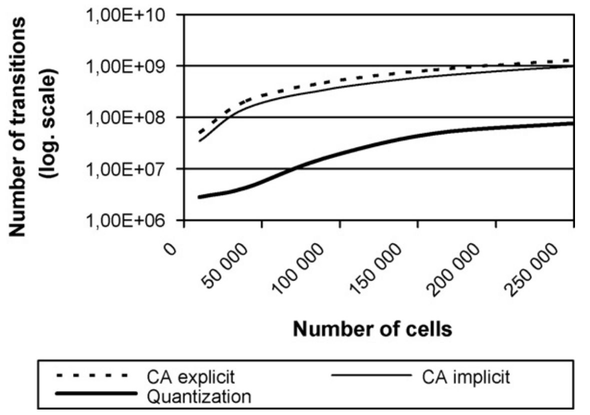


Fig. 15 – Number of transitions for different domain sizes (quantization model: $D = 5$ K, implicit CA: $\Delta t = 0.1$ s, explicit CA: $\Delta t = 0.01$ s).

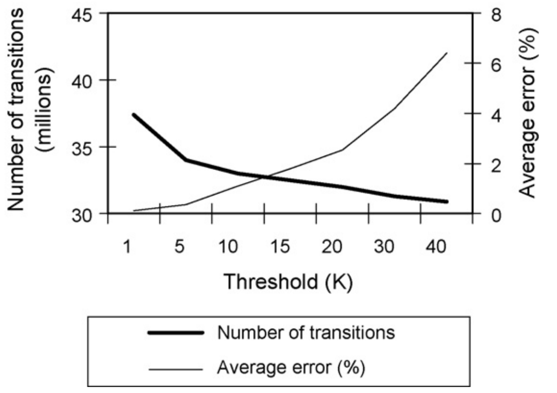


Fig. 16 – Number of transitions and error for different temperature quanta and 10,000 cells (implicit DSCA: $\Delta t = 0.1$ s).

3.3.2. *Explicit method in dynamic structure cellular automata*

Fig. 17 depicts the error of the DSCA explicit simulation. Here, surprising results are obtained. Indeed, until a temperature gradient of 10 K, errors of the restricted calculation domain balance those induced when all cells of the calculation domain compute their states. However, very small errors are induced. Fig. 17 depicts too the number of transitions (which is proportional to the execution time) for different threshold sizes when simulating DSCA explicit schemes. As for DSCA implicit schemes, we can notice that reducing the threshold size does not reduce the number of transitions as effectively as the quantization method. More over, when compare to the DSCA implicit scheme, the reduction of transitions decreases more slowly.

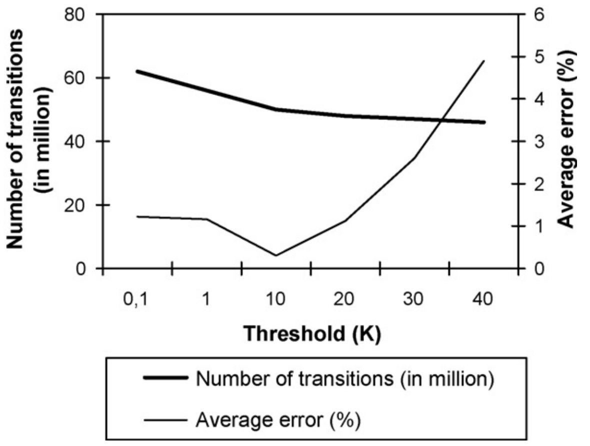


Fig. 17 – Number of transition and error for different quantum sizes and 10,000 cells (explicit DSCA: $\Delta t = 0.01$ s).

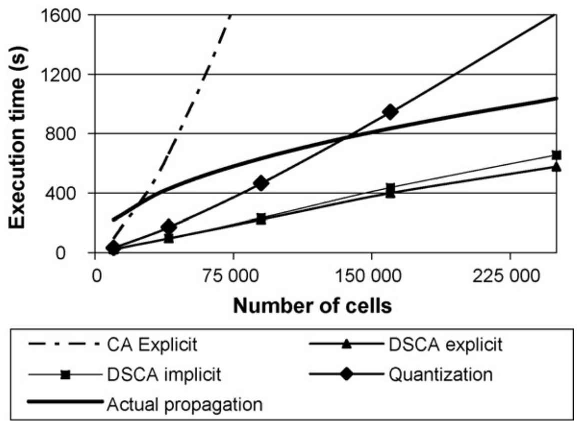


Fig. 18 – Execution time comparisons.

4. Discussion

For different numbers of cells, Fig. 18 compares execution times of all methods (except the CA implicit one which is approximately similar to the CA explicit one). To have approximately the same error, a quantum of 5 K is taken for the quantization method and thresholds of 10 K for both implicit and explicit DSCA ones. When compared to the actual propagation time, only the DSCA methods (explicit and implicit) give execution times that could satisfy a real time constraint. Indeed, in a simulator prediction perspective, it would be better to predict fire positions quicker than the fire propagates in the reality. Furthermore, both of the DSCA methods provide approximately the same execution time results.

Above 150,000 cells, the execution time of the quantization method exceeds the actual propagation time. Above this number, the number of active cells is too high to be managed by scheduler data structures of discrete-event approaches. This effect is shown in Fig. 18. Nonetheless, the quantization method necessitates less computations than the non-adaptive explicit and implicit methods.

Fig. 19 describes the rate of spread of the fire front. After the ignition, fire spread exhibit a transient and then a steady state corresponding to a constant rate of spread.

The evolution of the number of active cells during the simulation for different thresholds is presented in Fig. 20. We notice that even with a threshold of 1 K the number of active cells never exceeds 5000 cells i.e. less than one half of the total number of cells. Under 50 s fire speed grows quickly, this

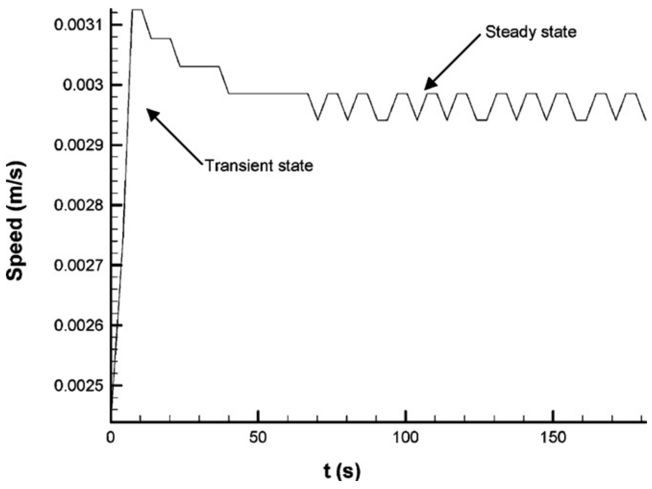


Fig. 19 – Fire front rate of spread.

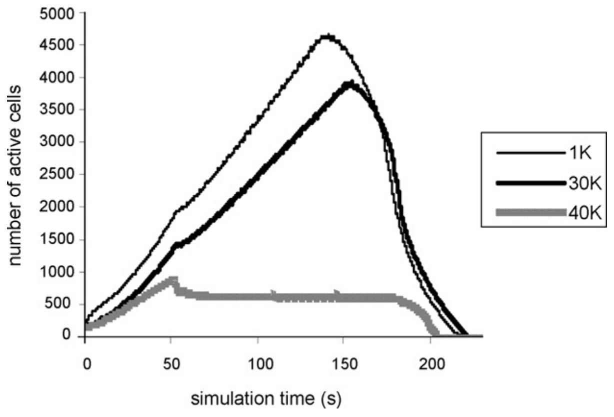


Fig. 20 – Number of active cells (explicit DSCA: Δt = 0.01 s).

is the transient state. Above 50 s, fire front speed reaches a steady state. This explains the change of slope at 50 s for the number of active cells. Above 150 s, the number of active cells decreases because fire reaches the plate borders.

Regarding the activity state of natural propagations the latter can be stationary or non-stationary (Coquillard, 1995). Qualitative and quantitative analysis of the activity of a natural phenomenon is a (or the?) fundamental (and usually neglected) modeling phase to understand the whole dynamics of an ecological system.

5. Conclusions

The activity paradigm has been applied here to a complex system of fire spread. Distribution of energy in space has been detected through well-defined structures. Complexity of energy exchanges and transformations has been defined modularly and hierarchically. This new paradigm proves to be well adapted to spatial ecological propagations (of trees, alga, fishes, etc.). Its genericity can be experimented now on other energy/information/material flows.

At the computer modeling and simulation levels, we have demonstrated that significant reductions of execution times can be obtained by naturally focusing computations on active cells of a large-scale cellular model. Two different methods have been introduced. The DSCA method allows the modeler to design tracking mechanisms at the modeling level, truncating information. On the other hand, the quantization method is a technique for discretizing the state of a continuous system. DSCA results proved to be better than the quantization ones for small errors, but the quantization method allows for qualitative simulation that preserves the fire shape and greatly reduces the execution time.

A new paradigm has been applied for building simulations that adaptively match their execution strategy to suit the dynamics of the model that is being simulated. Advantages of using this paradigm for modeling and simulating components of environmental phenomena can be summarized as follows:

1. A coherent formal and operational framework is provided to build computational environmental models of continuous and discrete systems. Both discrete-time and discrete-event driven constructions can be achieved.
2. Knowledge about environmental models can be organized within the DEVS-scheme (Vasconcelos, 1993), trough taxonomy, decompositions and couplings.
3. Tracking mechanisms are made simple and explicit. Direct, faithful and intuitive mappings between Nature and computers are defined.
4. Both simulation and models are made more efficient. A trade-off between error and execution time can be adjusted to fit modeling objectives.

The proposed techniques have significant potential that has been demonstrated empirically here and elsewhere (e.g., in simulation: Muzy and Nutaro, 2005 and formal modeling: Muzy et al., 2004); the method merits further empirical and formal investigation to verify, validate and extend its utility.

REFERENCES

Albini, F.A., 1985. A model for fire spread in wildland fuels by radiation. Combustion Science and Technology 42, 229–258.
Albright, D., Meisner, B.N., 1999. Classification of fire simulation systems. Fire Management Notes 59 (2), 5–12.
Alfredsen, K., Sæther, B., 2000. An object-oriented application framework for building water resource information and planning tools applied to the design of a flood analysis system. Environmental Modelling and Software 15 (3), 215–224.
Alonso-Sanz, R., 2007. A structurally dynamic cellular automaton with memory in the triangular tessellation. Complex Systems 17, 1–15.
Ameghino, J., Tróccoli, A., Wainer, G., 2001. Models of complex physical systems using Cell-DEVS. In: Annual Simulation Symposium (ANSS'01), Seattle, U.S.A, pp. 266–273.
Balbi, J.H., Santoni, P.A., Dupuy, J.L., 1998. Dynamic modelling of fire spread across a fuel bed. International Journal of Wildland Fire 9 (4), 275–284.
Balzter, H., Braun, P.W., Kohler, W., 1998. Cellular automata models for vegetation dynamics. Ecological Modelling 107, 113–125.
Barros, F.J., 1997. Modelling formalisms for dynamic structure systems. ACM Transactions on Modelling and Computer Simulation 7 (4), 501–515.

Barros, F.J., Mendes, M.T., 1997. Forest fire modelling and simulation in the DELTA environment. Simulation Practice and Theory 5 (3), 185–197.

Baveco, J.M., Smeulders, A.M.W., 1994. Objects for simulation: smaltalk and ecology. Simulation 62 (1), 42–57.

Berjak, S.G., Hearne, J.W., 2002. An improved cellular automaton model for simulating fire in a spatially heterogeneous Savanna system. Ecological Modelling 148, 133–151.

Bolduc, J.S., Vangheluwe, H., 2002. Expressing ODE Models as DEVS: Quantization Approaches. AI, Simulation and Planning in High Autonomy Systems. SCS, Lisboa, Portugal, pp. 163–169.

Coleman, J.R., Sullivan, A.L., 1996. A real-time computer application for the prediction of fire spread across the Australian Landscape. Simulation 67 (4), 230–240.

Coquillard, P., 1995. Simulation of the cyclical process of heathlands: induction of mosaic structures, evolution to irreversible states. Ecological modelling 80, 97–111.

Dolling, K., Chu, P.-S., Fujioka, F., 2005. A climatological study of the Keetch/Byram drought index and fire activity in the Hawaiian Islands. Forest Ecology and Management 133, 17–27.

Dong Hyun, K., Myung Bo, L., Kyo Sang, K., Si Young, L., 2006. Forest fire risk assessment through analyzing ignition characteristics of forest fuel bed. Forest Ecology and Management 234, S31.

Finney, M.A., 1995. FARSITE Fire Area Simulator Version 1.0 user guide and technical documentation. Missoula, MT.

Hargove, W.W., Gardner, R.H., Turner, M.G., Romme, W.H., Despain, D.G., 2000. Simulating fire patterns in heterogeneous landscapes. Ecological Modelling 135, 243–263.

Hernández Encinas, S. Hoya White, Martin del Rey, A., Rodriguez Sanchez, G., 2007. Modelling forest fire spread using hexagonal cellular automata. Appl. Math. Model. 31, 1213–1227.

Hill, D., Coquillard, P., Vaugelas, J.D., Meinez, A., 1998. An algorithmic Model for Invasive Species Application to Caulerpa taxifolia (Vahl) C Agardh development in the North-Western Mediterranean Sea. Ecological Modelling 109, 251–265.

Holst, N., Axelsen, J.A., Olesen, J.E., Ruggle, P., 1997. Object-oriented implementation of the metabolic pool model. Ecological Modelling 104, 175–187.

Karafyllidis, I., Thanailakis, A., 1997. A model for predicting fire spreading using cellular automata. Ecological Modelling 99, 87–97.

Lawrie, J., Hearne, J., 2007. Reducing model complexity via output sensitivity. Ecological Modelling 207 (2–4), 137–144.

Lopes, A.M.G., Cruz, M.G., Viegas, D.X., 2002. FireStation—an integrated software system for the numerical simulation of fire spread on complex topography. Environmental Modelling & Software 17 (3), 269–285.

McArthur, A.G., 1966. Weather and grassland fire behaviour. Australian Forest and Timber Bureau Leaflet 100.

Miller, J.D., Yool, S.R., 2002. Modeling fire in semi-desert grassland/oak woodland: the spatial implications. Ecological Modelling 153, 229–245.

Muzy, A., Aïello, A., Santoni, P.-A., Zeigler, B.P., Nutaro, J.J., Jammalamadaka, R., 2005a. Discrete event simulation of large-scale spatial continuous systems. In: International Conference on Systems, Man and Cybernetics (SMC), IEEE, Hawaii, USA, pp. 2991–2998.

Muzy, A., Innocenti, E., Hill, D., Santucci, J.F., 2003. Optimization of cell spaces simulation for the modelling of fire spreading. In: 36th Annual Simulation Symposium, Orlando, USA, IEEE/SCS/ACM, pp. 289–296.

Muzy, A., Innocenti, E., Hill, D.R.C., Aïello, A., Santucci, J.F., Santoni, P.A., 2004. Dynamic structure cellular automata in a fire spreading application. Chosen as on of the best papers of the conference. In: First International Conference on Informatics in Control, Automation and Robotics, Setubal, Portugal, IEEE/CSS/IFAC/ACM/AAAI/APPIA, pp. 143–151.

Muzy, A., Innocenti, E., Wainer, G., Aïello, A., Santucci, J.F., 2005b. Specification of discrete event models for fire spreading. Among the 50 most-frequently-read articles in SIMULATION. SIMULATION: Transactions of the Society of Modeling and Simulation International 81, 103–117.

Muzy, A., Nutaro, J.J., 2005. Algorithms for efficient implementation of the DEVS & DSDEVS abstract simulators. In: 1st Open International Conference on Modeling and Simulation (OICMS), Clermont-Ferrand, France, pp. 273–279.

Muzy, A., Zeigler, B.P., 2008. Introduction to the activity tracking paradigm in component-based simulation. The Open Cybernetics and Systemics Journal 2, 48–56.

Norby, R.J., Ogle, K., Curtis, P.S., Badeck, F.-W., Huth, A., Hurtt, G.C., Kohyama, T., Peñuelas, J., 2001. Aboveground Growth and Competition in Forest Gap Models: an analysis for studies of climatic change. Climatic Change 51 (3/4), 415–447.

Ntaimo, L., Zeigler, B.P., 2005. Integrating Fire Suppression into a DEVS Cellular Forest Fire Spread Model. In: Spring Simulation MultiConference, San Diego, CA, USA, pp. 48–54.

Núñez-Regueira, L., Proupín-Castiñeiras, J., Rodríguez-Añon, J.A., 2000. Design of risk index maps as a tool to prevent forest fires in the hill-side zone of Galicia (NW Spain). Bioresource Technology 73, 123–131.

Núñez-Regueira, L., Rodríguez-Añon, J.A., Proupín-Castiñeiras, J., 2001. Calculation of biomass indices as a tool to fight forest fires. Thermochimica Acta 378, 9–25.

Nutaro, J., 2003. Parallel Discrete Event Simulation with Applications to Continuous Systems. University of Arizona, Tucson.

Nutaro, J., Zeigler, B.P., 2007. On the stability and performance of discrete event methods for simulating continuous systems. Journal of Computational Physics 227 (1), 797–819.

Papajorgji, P., Beck, H.W., Braga, J.L., 2004. An architecture for developing service-oriented and component-based environmental models. Ecological Modelling 179 (1), 61–76.

Parker, D.C., Manson, S.M., Janssen, M.A., Hoffmann, M.J., Deadman, P., 2003. Multi-agent systems for the simulation of land-use and land-cover change: a review. Annals of the Association of American Geographers 93, 314–337.

Ralston, A., Rabinowitz, P., 2001. A First Course in Numerical Analysis. Dover Publications.

Richards, G.D., 1990. An elliptical growth model of forest fire fronts and its numerical solution. International Journal of Numerical Method Engineering 30, 1163–1179.

Rothermel, R.C., 1972. A Mathematical Model for Predicting Fire Spread in Wildland Fuels. USDA, Forest Service Research.

Santoni, P.A., 1996. Forest fire propagation: dynamic modeling and numerical resolution, validation on fuel bed fires (in French). Faculté des Sciences et Techniques. Corti, Università di Corsica.

Sequeira, R.A., Sharpe, P.J.H., Stone, N.D., El-Zik, K.M., Makel, M.E., 1991. Object-oriented simulation: plant growth and discrete organ to organ interactions. Ecological Modelling 58, 55–89.

Sibony, M., Mardon, J.C., 1988. Approximations et équations différentielles. Hermann.

Silvert, W., 1993. Object-oriented ecosystem modelling. Ecological Modelling 68, 91–118.

Spanou, M., Daoyi, C., 2000. An object-oriented tool for the control of point-source pollution in river systems. Environmental Modelling and Software 15 (1), 35–54.

Thompson, W.A., Vertinsky, I., Schreier, H., Bruce, A., Blackwell, B.A., 2000. Using forest fire hazard modelling in multiple use forest management planning. Forest Ecology and Management 134, 163–176.

Vasconcelos, M.J., 1993. Modeling spatial dynamic ecological processes with DEVS-scheme and geographic information systems. Arizona, p. 169.

Vasconcelos, M.J., Pereira, J.M.C., Zeigler, B.P., 1995. Simulation of fire growth using discrete event hierarchical modular models. EARSeL Advances in Remote Sensing 4 (3), 54–62.

Veach, M.S., Coddington, P.D., Fox, G.C., 1994. BURN: A Simulation of Forest Fire Propagation. Northeast Parallel Architectures Center.

Weber, R.O., 1990. Modelling fire spread through fuel beds. Progress in Energy and Combustion Science 17, 67–82.

Wu, J., David, J.L., 2002. A spatially explicit hierarchical approach to modelling complex ecological systems: theory and applications. Ecological Modelling 153, 7–26.

Wu, Y., Sklar, F.H., Gopu, K., Rutchey, K., 1996. Fire simulations in the Everglades Landscape using parallel programming. Ecological Modelling 93, 113–124.

Yassemia, S., Dragićević, S., Schmidt, M., 2008. Design and implementation of an integrated GIS-based cellular automata model to characterize forest fire behaviour. Ecological Modelling 210, 71–84.

Zeigler, B.P., 2005. Discrete Event Abstraction: An Emerging Paradigm for Modeling Complex Adaptative Systems. Adaptation and Evolution (festschrift for John H. Holland). E. Oxford Press, Santa Fe Institute.

Zeigler, B.P., Praehofer, H., Kim, T.G., 2000. Theory of Modelling and Simulation. Academic Press.