# Formal Agent-Based Modelling and Simulation of Crowd Behaviour in Emergency Evacuation Plans

Ioanna Stamatopoulou
The University of Sheffield
International Faculty, City College
Thessaloniki, Greece
Email: istamatopoulou@city.academic.gr

Ilias Sakellariou
Dept. of Applied Informatics
University of Macedonia
Thessaloniki, Greece
Email: iliass@uom.gr

Petros Kefalas
The University of Sheffield
International Faculty, City College
Thessaloniki, Greece
Email: kefalas@city.academic.gr

***Abstract*—Crowd behaviour deviates from normal when an emergency evacuation is needed. Thus, simulation of evacuation situations has been identified as an important tool for assessing design choices of urban areas, such as buildings, stadiums, etc., and Agent Based Modelling has been employed to tackle such problems. In this paper, we propose that formal modelling can rigorously define but also naturally lead to realistic simulations of such cases. Our main contribution is presenting how formal state-based methods, namely X-machines, can be employed to model agents in emergency evacuation plans. We also discuss the role of emotions, model artificial emotions that change the behaviour of agents under emergency situations, and provide a formalism that models the role of emotions and personality traits in order to create a more realistic scenario. Finally, we demonstrate how the developed formal models can be refined to code, a combination of Netlogo and Prolog in this case, that is able to simulate crowd behaviour with and without artificial emotions.**

## I. Emergency Evacuation as a Multi-Agent System

### A. Evacuation Analysis and Simulation

A crucial aspect to the design of modern urban areas, such as buildings, stadiums, metro stations, etc., is evacuation analysis, i.e. assessing the the evacuation capability of the area under emergency conditions (fire, earthquake, etc.). Under such situations crowd behaviour deviates from normal in the sense that the crowd immediately proceeds towards the exits, there is panic and pushing, resulting in many cases in injuries and fatalities. Computer based simulation of evacuation situations has been identified as an important tool for assessing design choices, with a large number of evacuation models proposed [1] that follow different approaches with respect to the method used and most importantly the granularity of the simulation, also referred to as scale.

A significant part of research takes the stance that crowd movement and *fluid dynamics* present remarkable similarities. Thus, flow dynamics are used, as in [2], in order to assess designs and simulate crowd behavioural patterns moving in an area of study. These models follow a "macroscopic" model, describe crowd behaviour as a set of partial differential equations, and are most useful in densely populated environments.

In *social force* models [3], pedestrians are considered as entities exposed to "force field" generated by individual intentions, other pedestrians and points of attraction. Models in this category have been thoroughly studied, adapted to model crowds in panic, and combined with agent based approaches [4], to successfully model crowd behaviour. Social force models follow the "microscopic" approach according to which the overall behaviour is an emergent property of interactions of individual entities.

*Cellular automata* [5], [6] describe the world as a grid of cells, while behaviour is determined by a set of local rules that update the state of each cell based on the state of its neighbouring cells. Gas lattice models consider pedestrians as "particles" moving on a grid with a set of probability rules determining their next position [7].

### B. Multi-Agent System for Crowd Behaviour

*Agent Based Modelling* (ABM) has also been employed to tackle the problem of crowd behaviour. ABM has a number of advantages [8], as it is flexible, allows a natural description of the model and emergent phenomena to manifest. Since ABM models can naturally model the diversity of a population, they seem to offer a valuable tool for simulating evacuation scenarios. For instance in [9] ABM is used for the crowd behaviour and cellular automata to model the environment, in a case of emergency evacuation in a metro station. In [10] emotional agents are used to create a simulation of an actual aircraft accident and validate the results against real life data.

The significant amount of published work in the area of evacuation modelling and simulation could not possibly be reported in the context of the present paper; there is a significant amount of research work and tools addressing the problem and the reader should refer to [1], [11], [12] for a more in depth review and assessment of proposed models.

Building on the idea that ABM is well suited to emergency evacuation scenarios, we propose that formal modelling can rigorously define and naturally lead to realistic simulations of such cases. The main aim of this paper is to show how formal state-based methods can be employed to model artificial agents on emergency evacuation plans, and to demonstrate how formal models can be refined to code that is able to simulate crowd behaviour under such situations. Finally, we provide a formalism that models the role of emotions and personality traits in order to make a more realistic simulation scenario.

The structure of this paper is as follows: Section II presents the X-machines formalism, the emergency evacuation scenario

under study, and the defined model of the participating rational agents. Section III presents the developed Netlogo simulation and discusses how it has been derived from the formal model. Section IV discusses the role of emotions, presents an extension to the formalism, so as to support the modelling of agents behaving under artificial emotions, the resulting simulation and some comparative results. Finally, Section V concludes the paper and suggests direction for future work.

## II. Formal Modelling of State-Based Agents in an Evacuation Scenario

Modelling is a stage in agent system development in which the attributes of an agent are described in an adequately abstract way to be useful for implementation. Formal modelling implies the use of mathematical notation and although it can be viewed as a pedantic step towards development, it offers a number of advantages that informal or semi-formal notations do not, such as the ability to prove correctness with respect to the specification. Correctness can be achieved by the process of verifying that certain properties hold in the original model and that the system implementation behaves in the right way under a complete test set.

Both formal verification and testing are computationally intensive processes. In particular types of agent systems, such as spatial multi-agent systems (MAS) representing crowd behaviour, formal proofs run into combinatorial explosion. On the other hand, model checking as formal verification would require the properties to be verified to be known in advance, which cannot always be true, especially in MAS with emergent behaviour. This is where simulation and visualisation play an important role towards identifying visual or statistical patterns of the overall behaviour of the system.

### A. An Emergency Evacuation Plan

The case that we choose to model has to do with a building evacuation in an emergency situation. More specifically, we attempt to model the behaviour of individual people from the point that danger is perceived until their exit from the building. We also take into consideration that some kind of an evacuation plan is available and may be followed by the individuals. This plan is usually of the form of a floor blueprint that is posted at various spots, indicating the current position of the person looking at the blueprint and the path to the exit. The path for the purposes of this work is considered to be a sequences of points in space (milestones) that progressively brings the individual closer to the exit (Fig. 1).

### B. Modelling with X-machines

There exist numerous formal methods, either general or specialised to agent modelling [13], [14]. We chose to work with X-machines (XM), which are state-based machines extended with a memory structure. The memory structure, which is an n-tuple of values, makes the machine more compact compared to memory-less state machines. Another important difference is that the transitions between states are not triggered by inputs alone, but by functions that accept an input and the memory values and produce an output and new memory values.

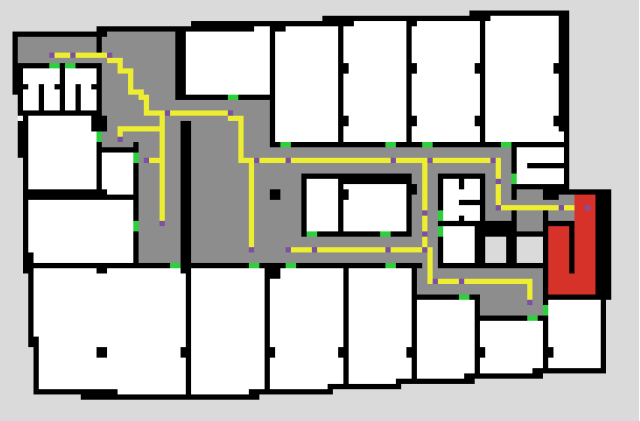

Fig. 1. Example blueprint of a building floor. White areas represent rooms and the grey area is the corridor. The plan to the exit is indicated by the squares in the corridor that are connected with a line, and lead to the stairs (exit) to the right.

*Definition 1:* A *stream X-machine* [15] is an 8-tuple

$$\mathcal{X} = (\Sigma,\ \Gamma,\ Q,\ M,\ \Phi,\ F,\ q_0,\ m_0)$$

where:

- $\Sigma$ and $\Gamma$ are the input and output alphabets, respectively.
- $Q$ is the finite set of states.
- $M$ is the (possibly) infinite set called memory.
- $\Phi$ is a set of partial functions $\varphi$; each such function maps an input and a memory value to an output and a possibly different memory value, $\varphi : \Sigma \times M \rightarrow \Gamma \times M$.
- $F$, often described as a state transition diagram, is the next state partial function, $F : Q \times \Phi \rightarrow Q$, which given a state and a function from $\Phi$ determines the next state.
- $q_0$ and $m_0$ are the initial state and memory respectively.

*Definition 2:* A computation state is defined as the tuple $(q, m)$, with $q \in Q$ and $m \in M$. The computation step is defined as $(q, m) \overset{\varphi}{\vdash} (q', m')$ with $q, q' \in Q$ and $m, m' \in M$ such that $\varphi(\sigma, m) = (\gamma, m')$ and $F(q, \varphi) = q'$. The computation is the series of computation steps when all inputs are applied to the initial computation state $(q_0, m_0)$.

X-machines are particularly suited for the modelling of reactive agents, as they make the modelling process more intuitive. It has been demonstrated that XM and its extensions are particularly useful for modelling biological and biology-inspired MAS [16]. The great advantage over other methods is their strong legacy of theory and practice in: (a) modelling potential for dynamically structured MAS [17], (b) refinement, animation and simulation [18], (c) testing methods that prove correctness [19] with tools for automatic test generation [20], and (d) model checking for verification of properties [21].

To model the given evacuation scenario using XM we initially consider that all individuals are placed *in* the rooms, same way students are in the classrooms during lectures, when an emergency is perceived. The memory of the XM holds

the evacuation plan (initially empty denoted by $\varepsilon$) and the current position of the person. The plan is considered as a sequence of coordinates-milestones that have to be reached, that lead to the exit. Therefore, $Plan : seq(Position)$, where $Position : \mathbb{N} \times \mathbb{N}$, and $m_0 = (\varepsilon, Pos)$.

The input to the XM at any one point is a set of elements that describe the individual's percepts and is of the form: $\Sigma = PosDescriptions \cup \{danger,\ checkIfReached,\ checkIfAtExit, dissorientation\}$ where $PosDescriptions : Position \times \{empty, door\}$.

Upon perceiving the danger an individual starts looking for the plan posted next to the room door and therefore starts moving towards it. When the plan is read the person proceeds with the evacuation process by moving in the surrounding available space closer to the next point that is indicated by the plan (first coordinate pair in the sequence). When such a point is reached it is removed from the head of the list and the individual starts moving towards the next one.

The $dissorientation$ input element is for modelling that in some cases an individual in distress gets dissoriented, which in this case means that the plan is lost; the person does not know where to go and starts wandering until another plan is found. $checkIfReached$ and $checkIfAtExit$ are there to model that the individual is constantly "checking" whether the next point in the plan or the exit has been reached.

The state transition diagram of the model is depicted in Fig. 2 and we present below some of the functions involved for indicative purposes.

$\varphi_{perceiveDanger} : (Percept, (\varepsilon, Pos)) \mapsto$
$(\text{“}DangerPerceived\text{”}, (\varepsilon, Pos))$,
if $danger \in Percept$

$\varphi_{searchForPlan} : (Percept, (\varepsilon, Pos)) \mapsto$
$(\text{“}LookingForPlan\text{”}, (\varepsilon, NewPos))$,
*if* $(NewPos, empty) \in Percept$
*and* $neighbours(Pos, NewPos)$
*and* $(Plan, plan) \notin Percept$

$\varphi_{moveToNext} : (Percept, (NextPoint :: Rest, Pos)) \mapsto$
$(\text{“}FollowingPlan\text{”}, (NextPoint :: Rest, NewPos))$,
*if* $(NewPos, empty) \in Percept$
*and* $neighbours(Pos, NewPos)$
*and* $closer(Pos, NextPoint, NewPos)$

$\varphi_{getDissoriented} : (Percept, (Plan, Pos)) \mapsto$
$(\text{“}GotDissoriented\text{”}, (\varepsilon, Pos))$,
*if* $dissorientation \in Percept$

## III. Simulation and Visualisation

Simulation is a valuable tool for informally verifying MAS. Particularly in cases of spatial properties, the visualisation provided by a simulation platform can reveal a variety of desired or unwanted/unexpected emerging behaviours.

NetLogo [22] has been widely accepted as a tool for agent based simulation of emerging and social phenomena. It offers an easy to use environment for the development of the simulation experiment with strong visualization facilities, and a domain specific language with a strong functional flavour.

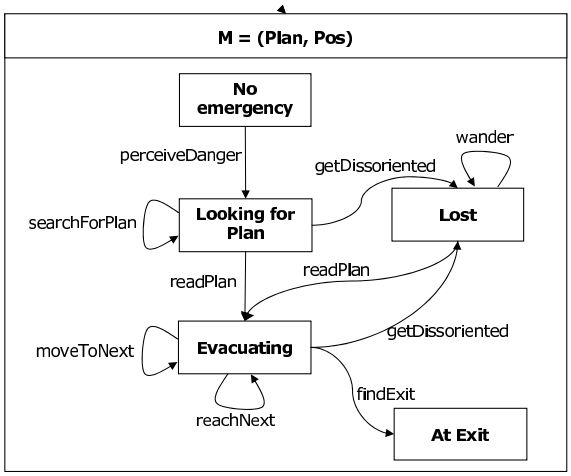


Fig. 2. The XM state transition diagram.

Direct encoding, however, of a stream XM model is not supported by the NetLogo language. Thus a modified version of a state machine DSL was developed that supports stream XMs [23]. The new DSL allows a direct encoding of the transition diagram depicted in Fig. 2, in the following form (part of the actual implementation):

```
to-report state-def-of-persons
report (list
 state "No Emergency"
 # x-func "perceiveDanger" goto "Looking for Plan"
 # otherwise do "nothing" goto "No Emergency"
 end-state

 state "Evacuating"
 # x-func "findExit" goto "At Exit"
 # x-func "reachNext" goto "Evacuating"
 # x-func "moveToNext" goto "Evacuating"
 # otherwise do "nothing" goto "Evacuating"
 end-state
```

An execution layer is responsible for "running" the agents, providing input from the simulation environment and updating the agent simulation state. Although the programming facilities and the execution layer were developed in NetLogo's own language, XM function semantics rely heavily on notions such as argument unification, and thus were encoded in Prolog and invoked through a NetLogo-Prolog language interface, called NetPrologo[1]. Following this approach, the model's functions enjoy direct encoding in Prolog. Given the above, the simulation environment follows the architecture depicted in Fig. 3 and, for instance, the function:

$\varphi_{perceiveDanger} : (Percept, (\varepsilon, Pos)) \mapsto$
$(\text{“}DangerPerceived\text{”}, (\varepsilon, Pos))$, *if* $danger \in Percept$

is directly encoded in Prolog as:

```
perceiveDanger(Percept, [ [], Pos ],
               dangerPerceived, [ [], Pos] ):-
member(danger, Percept).
```

In the experiment presented the following assumptions hold:

[1]NetPrologo is available at http://www.cs.us.es/ fsancho/NetProLogo/

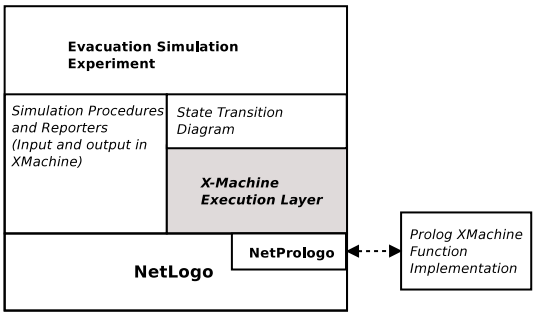


Fig. 3. Architecture of the Simulator.

- Space is discrete, with the common assumption that each individual is occupying a 40×40 cm cell and can move to eight possible neighbouring positions.
- All students are inside the classrooms and, upon hearing the alarm, move towards the door of the room, where they can read the evacuation plan for their specific location. Thus, in each door a different evacuation plan is advertised.

Note, at this point, that a number of NetLogo reporters were implemented that provide input to the XM execution layer, as well as code for updating the agents' position in the world according to the output of the XM.

Having the layer for specification and execution of stream XM agents, we have developed a simulation experiment (Fig. 4). The experiment allows a number of parameters to be set, such as the total number of people on the floor at the time of the alarm, as well as the monitoring of various parameters, such as the average and total evacuation times.

Although the environment could be more detailed, e.g. by including desks, chairs, etc., such a world model was considered outside the scope of this work, its focus being on demonstrating and emphasizing the agent modelling approach.

## IV. The role of Emotions

In order to achieve a more realistic simulation, agents must be able to demonstrate believable behaviour, modelling humans who under stress or panic make irrational decisions. Emotions influence agent perception, learning, behaving, communication, etc., and so far, there is no widely accepted theory describing how emotional processes affect reasoning in general [24]. A number of computational models of emotions logically formalised as BDI agents have been attempted [25]. The role of emotions as well as the type of agents in emergency evacuation has been receiving increased attention [26].

We chose to not to adopt any particular emotion theory but to consider basic emotions, referred to as "artificial emotions", which are plugged-in to the XM definition in order to facilitate modelling of emotional agents.

### A. Modelling with $^{e}$X-machines

*Definition 3:* An *emotions X-machine* is defined as [27]:

$$^{e}\mathcal{X} = (\Sigma,\ \Gamma,\ Q,\ M,\ \Phi,\ F,\ q_0,\ m_0,\ E,\ ^{e}\Phi,\ e_0)$$

where:

- $\Sigma$, $\Gamma$, $Q$, $M$, $F$, $\Phi$, $q_0$ and $m_0$ are as in *stream X-machines*
- $E = (\epsilon_1, \ldots, \epsilon_n)$ is a vector of emotion identifiers.
- $^{e}\Phi : E \times M \times \Sigma \rightarrow E \times M$ is the emotion revision function.
- $e_0$ is the initial vector of emotion identifiers representing the initial emotional state.

An $^{e}\mathcal{X}$ model has a computational state as well an emotional state represented by the vector of emotional identifiers. An input triggers the emotion revision function, changing the emotional state and memory, as well as a transition function, returning a new state. It is therefore possible that emotions change the computation path by allowing bias towards one or another function. The computation is defined as follows.

*Definition 4:* A computation state is defined as the tuple $(q, m, e)$, with $q \in Q$ and $m \in M$ and $e \in E$. The computation step, which consumes an input $\sigma \in \Sigma$ and changes the computation state $(q, m, e) \vdash (q', m', e')$ is composed of two substeps:

- $(q, m, e) \overset{^{e}\varphi}{\vdash} (q, m'', e')$ with $q \in Q$, $e, e' \in E$ and $m, m'' \in M$, such that $^{e}\varphi(e, m, \sigma) = (e', m'')$
- $(q, m'', e') \overset{\varphi}{\vdash} (q', m')$ with $q, q' \in Q$, $m', m'' \in M$ and $e' \in E$, such that $\varphi(\sigma, m'', e') = (\gamma, m')$ and $F(q, \varphi) = q'$.

The computation is defined as the series of computation steps that take place when all inputs are applied to the initial computation state $(q_0, m_0, e_0)$.

Bearing in mind the definition of the $^{e}$XM, the model developed in Section II-B is now enhanced with an emotional state. We include the basic emotion *Horror* [28] that can be assigned a set of crisp values: $HorrorLevel = \{calm, alarm, fear, terror, panic, hysteria\}$. To model the strength of the emotion and determine when a horror level is reached, a value $S = 0 \ldots 100$ is required. Consequently, an emotion revision function $emotionRevision : (HorrorLevel, S) \rightarrow (HorrorLevel, S)$ can update the strength and the horror level, given the current values, and the emotional vector is now defined as $E = (HorrorLevel, S)$, with $e_0 = (calm, 0)$.

The rate of change of $E$ may remain the same for all agents but in more realistic situations it depends on individual agents. Some individuals have a predisposition towards experiencing certain emotions, and different *personality traits* are responsible for how quickly an emotional state is reached, maintained and recovered from [29]. Therefore, some agents reach a state of panic more easily than others who remain calm or alarmed for a longer period. To conclude, the XM emotion revision function $^{e}\varphi$ for the given scenario is defined as follows:

$^{e}\varphi : (Percept, (Plan, Pos, PersonTrait), (T, S)) \mapsto$
$((Plan, Pos, PersonTrait), (T', S'))$,
*where* $(T', S') \leftarrow emotionRevision(PersonTrait, S)$

where $emotionRevision$ is defined so as to take into consideration the personality trait of a person and the current strength


Fig. 4. NetLogo Simulation of a building emergency evacuation scenario in which agents behave rationally (without emotions).

of the emotion and increase or decrease the latter accordingly.

A concrete instance of an emotion revision can be seen below where a calm person becomes alarmed when danger is initially perceived :

${}^{e}\varphi_{getAlarmed} : (Percept, (\varepsilon, Pos, PersonTrait), (calm, 0))$
$\mapsto ((\varepsilon, Pos, PersonTrait), (alarm, S'))$,
*if* $danger \in Percept$,
*where* $(alarm, S') \leftarrow emotionRevision(PersonTrait, 0)$

So, $emotionRevision$ will increase the strength of the artificial emotion to a value biased proportionally to the personality trait of the agent.

### B. Simulation Results under Emotional Reactions

Based on the revised ${}^{e}$XM model, we developed a more realistic simulation (Fig. 5). Both snapshots in Figures 4 and 5 were taken 50 cycles after the alarm sound, and it is evident that the overall behaviour is affected. In the first case, the crowd is well formed, walking towards the exit, while in the second it is more dispersed; the number of agents who are lost is increased due to the fact that they reach a state of hysteria that prevents them from following the evacuation plan. The difference is also evident in Table I where the total evacuation times are listed for both cases, with various numbers of agents in the building. It is interesting to spot that sometimes a bigger number of agents does not necessarily lead to a greater total evacuation time or greater individual average. This is due to a critical mass of agents that "push" other lost agents to towards the exit. Further analysis of such results does not fall in the scope of this paper, in which we aim only to demonstrate how formal modelling can lead to a visual simulation.

TABLE I
TOTAL EVACUATION TIME IN ALTERNATIVE SCENARIOS; AVERAGE INDIVIDUAL EVACUATION TIME IN BRACKETS

| Number of Agents | Rational Behaviour | Behaviour with Emotions |
|---|---|---|
| 100 | 2600 (311) | 9300 (880) |
| 200 | 3380 (281) | 8270 (960) |
| 300 | 3914 (312) | 13200 (1000) |
| 400 | 4360 (385) | 15424 (1221) |

## V. Conclusions and Further Work

In this paper, we have demonstrated how formal agent based modelling can be used for crowd behaviour in emergency evacuation scenarios, using X-machines and its extension ${}^{e}$X-machines. The behaviour of each agent was formally modelled twice; once acting totally rationally and once acting under artificial emotions. Respectively, two different visual simulations were developed based on the formal models. NetLogo in collaboration with Prolog, was shown to suit the task perfectly. Results verified the expected difference in overall behaviour, depicting the difference between an ideal and a realistic scenario.

The artificial emotions model is by no means complete. It needs further enhancement to deal with emotions that affect perception appraisal, communication, etc., and the agents can be further classified according to more factors affecting crowd behaviour, such as having families and other social interactions.

Although the NetPrologo interface provides an elegant solution to the problem of encoding XM functions in the simulation, it imposes a computational overhead in the execution of the model. Thus, one main direction towards the simulation of larger scale experiments will be to develop a compiler from XM specifications to NetLogo that will offer faster execution of the simulation experiments.

## References

[1] G. Santos and B. E. Aguirre, "A critical review of emergency evacuation simulation models," in *Workshop on Building Occupant Movement*

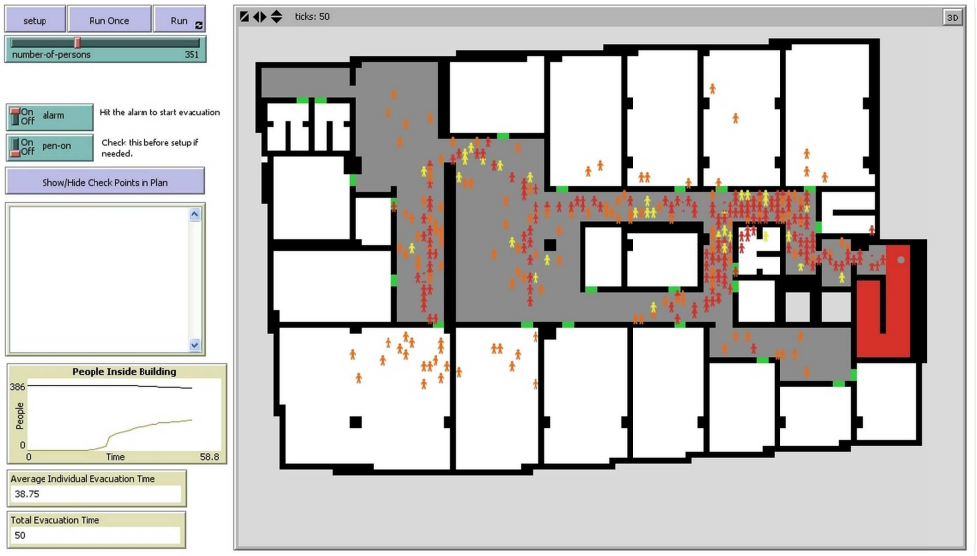


Fig. 5. NetLogo Simulation of a building emergency evacuation scenario in which agents behave under artificial emotions.

*During Fire Emergencies, 2004*, R. D. Peacock and E. D. Kuligowski, Eds. NIST Special Publication 1032, 2005, pp. 27–52.

[2] R. L. Hughes, "A continuum theory for the flow of pedestrians," *Transportation Research Part B: Methodological*, vol. 36, no. 6, pp. 507–535, 2002.

[3] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical Review E*, vol. 51, p. 4282, 1995.

[4] Q. Lin, Q. Ji, and S. Gong, "A crowd evacuation system in emergency situation based on dynamics model," in *Interactive Technologies and Sociotechnical Systems*, ser. LNCS, H. Zha, Z. Pan, H. Thwaites, A. Addison, and M. Forte, Eds. Springer, 2006, vol. 4270, pp. 269–280.

[5] V. Blue and J. Adler, "Cellular automata microsimulation for modeling bi-directional pedestrian walkways," *Transportation Research Part B: Methodological*, vol. 35, no. 3, pp. 293–312, 2001.

[6] H. Yue, H. Hao, X. Chen, and C. Shao, "Simulation of pedestrian flow on square lattice based on cellular automata model," *Physica A: Statistical Mechanics and its Applications*, vol. 384, no. 2, pp. 567–588, 2007.

[7] R. Guo and H. J. Huang, "A mobile lattice gas model for simulating pedestrian evacuation," *Physica A: Statistical Mechanics and its Applications*, vol. 387, no. 2-3, pp. 580–586, Jan 2008.

[8] E. Bonabeau, "Agent-based modeling: Methods and techniques for simulating human systems," *Proceedings of the National Academy of Sciences of the USA*, vol. 99, no. Suppl 3, pp. 7280–7287, 2002.

[9] N. Zarboutis and N. Marmaras, "Searching efficient plans for emergency rescue through simulation: the case of a metro fire," *Cognition, Technology and Work*, vol. 6, pp. 117–126, 2004.

[10] T. Miyoshi, H. Nakayasu, Y. Ueno, and P. Patterson, "An emergency aircraft evacuation simulation considering passenger emotions," *Computers & Industrial Engineering*, vol. 62, no. 3, pp. 746–754, 2012.

[11] S. Zhou, D. Chen, W. Cai, L. Luo, M. Y. H. Low, F. Tian, V. S.-H. Tay, D. W. S. Ong, and B. D. Hamilton, "Crowd modeling and simulation technologies," *ACM Trans. Model. Comput. Simul.*, vol. 20, no. 4, pp. 1–35, 2010.

[12] X. Zheng, T. Zhong, and M. Liu, "Modeling crowd evacuation of a building based on seven methodological approaches," *Building and Environment*, vol. 44, no. 3, pp. 437–445, 2009.

[13] F. F. Ingrand, M. P. Georgeff, and A. S. Rao, "An architecture for real-time reasoning and system control," *IEEE Expert: Intelligent Systems and Their Applications*, vol. 7, no. 6, pp. 34–44, 1992.

[14] S. J. Rosenschein and L. P. Kaelbling, "A situated view of representation and control," *Artificial Intelligence*, vol. 73, pp. 149–173, 1995.

[15] M. Holcombe and F. Ipate, *Correct Systems: Building a Business Process Solution*. London: Springer, 1998.

[16] P. Kefalas, I. Stamatopoulou, I. Sakellariou, and G. Eleftherakis, "Transforming Communicating X-machines into P Systems," *Natural Computing*, vol. 8, no. 4, pp. 817–832, December 2009.

[17] I. Stamatopoulou, P. Kefalas, and M. Gheorghe, "OPERAS: a formal framework for multi-agent systems and its application to swarm-based systems," in *Proceedings of the 8th International Workshop on Engineering Societies in the Agents World (ESAW'07)*, A. Artikis, G. O'Hare, K. Stathis, and G. Vouros, Eds., 2007, pp. 208–223.

[18] I. Petreska, P. Kefalas, and M. Gheorghe, "A framework towards the verification of emergent properties in spatial multi-agent systems," *Proceedings of the Workshop on Applications of Software Agents*, pp. 37–44, 2011.

[19] F. Ipate and M. Holcombe, "An integration testing method that is proved to find all faults," *International Journal of Computer Mathematics*, vol. 63, no. 3, pp. 159–178, 1997.

[20] D. Dranidis, K. Bratanis, and F. Ipate, "JSXM: A tool for automated test generation," in *The 10th International Conference on Software Engineering and Formal Methods*. Springer, 2012, to appear.

[21] G. Eleftherakis, "Formal verification of X-machine models: Towards formal development of computer-based systems," Ph.D. dissertation, Department of Computer Science, University of Sheffield, 2003.

[22] U. Wilensky, *NetLogo*, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL., 1999, http://ccl.northwestern.edu/netlogo/.

[23] I. Sakellariou, "Turtles as state machines—agent programming in netlogo using state machines," in *ICAART 2012, Proceedings of the 4th Int. Conf. on Agents and Artificial Intelligence, Volume 2 - Agents*, J. Filipe and A. L. N. Fred, Eds. SciTePress, 2012, pp. 375–378.

[24] P. Kleinginna and A. Kleinginna, "A categorized list of emotion definitions, with suggestions for a consensual definition," *Motivation and Emotion*, vol. 5, pp. 345–379, 1981.

[25] H. Jiang, J. M. Vidal, and M. N. Huhns, "EBDI: an architecture for emotional agents," in *Proc. of the 6th Int. Joint Conf. on Autonomous Agents and Multiagent Systems*. ACM, 2007, pp. 1–3.

[26] A. Zoumpoulaki, N. Avradinis, and S. Vosinakis, "A multi-agent simulation framework for emergency evacuations incorporating personality and emotions," in *AI: Theories, Models and Applications*, ser. LNCS, S. Konstantopoulos, S. Perantonis, V. Karkaletsis, C. Spyropoulos, and G. Vouros, Eds. Springer, 2010, vol. 6040, pp. 423–428.

[27] P. Kefalas, I. Stamatopoulou, and D. Basakos, "Formal modelling of agents acting under artificial emotions," in *Proceedings of the 5th Balkan Conference in Informatics (BCI2012)*, sep. 2012.

[28] W. Parrott, *Emotions in Social Psychology*. Psychology Press, Philadelphia, 2001.

[29] T. Dalgleish and M. Power, *Handbook of Cognition and Emotion*. Eds. Chichester: John Wiley and Sons, 1999.