GP Generation of Pedestrian Behavioral Rules in an Evacuation Model Based on SCA

Stefania Bandini, Sara Manzoni, Giancarlo Mauri, Stefano Redaelli, and Leonardo Vanneschi

Complex Systems and Artificial Intelligence Research Center
University of Milan–Bicocca, Italy
Tel.: +39-02-64487808
{bandini,manzoni,redaelli,vanneschi}@disco.unimib.it

Abstract. This paper presents a research in the context of pedestrian dynamics according to Situated Cellular Agent (SCA), a Multi-Agent Systems approach whose roots are on Cellular Automata (CA). The aim of this work is to apply Genetic Programming (GP) approach, a well known Machine Learning method belonging to the family of Evolutionary Algorithms, to generate suitable behavioral rules for pedestrians in an evacuation scenario. The main contribution of this work is in the design of a testset of GP generated behaviors to represent basic behavioral models of evacuees populating a only locally known environment, a typical scenario for CA-based models.

1 Introduction

Within pedestrian dynamics Cellular Automata approaches have found an interesting and fruitful application context (for instance in evacuation studies and public spaces' design [1,2]). Since relatively recent first proposals [11,12], CA approach has rapidly grown and shown interesting results concerning the study of potentially complex behaviors that can result from local interactions among pedestrians within a shared, limited, and only partially known spatial environment[3,4]. According to CA peculiarities the spatial environment can be represented as a regular grid of cells, whose state can include the representation of the presence of individuals (or other environmental obstacles). Pedestrian movement is represented by CA state transition rules and the dynamics of the system result from local state change of CA cells.

When adopting CA for modelling purposes with the aim of study pedestrian behavior and their interaction, CA models suffer, like traditional analytical approaches, the limitation of considering individuals as homogeneous entities whose behavior is implicitly represented in CA cell state and state transition function. SCA4CROWDS [5] is an ongoing research aiming at developing formal and computational tools to support the design, execution and analysis of models to study potentially complex dynamics that can emerge in crowds (e.g. pedestrian dynamics as effect of physical, social and emotional interactions) [6,23]. SCA4CROWDS formal model has been developed as an extension of Cellular Automata (CA)

H. Umeo et al. (Eds): ACRI 2008, LNCS 5191, pp. 409–416, 2008.

[7] exploiting MAS [8] advantages in modeling heterogeneous systems [9,10]. Despite models based on CA, SCA4CROWDS models systems of reactive situated agents (i.e. pedestrians) that move on structured spatial environments, and that can interact at-a-distance through the emission-diffusion-perception of signals and locally according to local transition rules. Theoretical experimentations are being developed with SCA4CROWDS formal and computational tools in order to provide methodological guidelines for sounding computational models of psychological and anthropological theories on crowds (e.g.[13]).

The work presented in this paper concerns the integration into SCA4CROWDS framework of formal and related computational tools to effectively combine Genetic Programming (GP) [14] with SCA approach to study the behavior of pedestrians (a similar approach has already been proposed for the calibration of a CA–based model of costumers in shopping areas [18]). In particular, we will present here a model based on SCA principles where pedestrian behaviors are based on GP, a well known evolutionary approach which extends the genetic model of learning to the space of programs. The best set of behavioral rules in an evacuation context drives a system of pedestrians evacuating a structured and unknown environment. Evacuation scenarios are traditionally considered realistic when they assume that pedestrians have a limited and very local information about the environment and a basic intelligent behavior that can be originated by instinctive or learning processes. To support the development and experimentation of this type of scenario, we developed a behavioral model based on a set of rules generated by Artificial Ant on the Santa Fe trail, a benchmark problem in GP [14]. The latter aims at finding navigation strategy for an ant moving on a regular grid (where some of the cells contain food pellets) that maximizes its food intake. This problem specification has been used as a methodological example useful for the SCA–based model calibration.

After an overview of SCA approach for pedestrian and crowds modeling, Genetic Programming approach will be introduced in Section 3, while Section 4 describes experiments with SCA4CROWDS framework where pedestrian behavioral rules of a set of evacuees are generated by GP. Currently we are developing an analytical analysis with references and benchmarks in pedestrian dynamics.

2 SCA Approach to Pedestrian Dynamics

According to SCA modelling approach, human crowds are described as system of autonomous, situated agents that act and interact in a spatially structured environment. Situated agents are defined as reactive agents that, as effect of the perception of environmental signals and local interaction with neighboring agents, can change either their internal state or their position on the structured environment. Agent autonomy is preserved by an action–selection mechanism that characterizes each agent, and heterogeneous MAS can be represented through the specification of agents with several behavioral types through L*MASS formal language [20] (an execution environment for SCA-based models is also available, i.e. SCA platform [19]). Interaction between agents can occur either locally,

causing the synchronous change of state of a set of adjacent agents, and at–a–distance, when a signal emitted by an agent propagates throughout the spatial structure of the environment and is perceived by other situated agents (heterogeneous perception abilities can be specified for SCA agents). Interaction primitives are defined by L*MASS language.

SCA model is rooted on basic principles of CA: it intrinsically includes the notions of state and explicitly represents the spatial structure of agents' environment; it takes into account the heterogeneity of modelled entities and provides original extensions to CA (e.g. at–a–distance interaction). According to SCA framework, the spatial abstraction in which the simulated entities are situated (i.e. *Space*) is an undirected graph of *sites* (i.e. $p \in P$), where graph nodes represent available space locations for pedestrians and graph edges define the adjacency relations among them (and agents' suitable movement directions). Each $p \in P$ is defined by $\langle a_p, F_p, P_p \rangle$, where $a_p \in A \cup \{\perp\}$ is the agent situated in p , $F_p \subset F$ is the set of fields active in p and $P_p \subset P$ is the set of sites adjacent to p. Pedestrians and relevant elements of their environment that may interact with them and influence their movement (i.e. *active elements of the environment*) are represented by different types of SCA agents that can change their internal state ($s \in \Sigma_\tau$), move into an adjacent site or interact with other agents. An agent *type* $\tau = \langle \Sigma_\tau, Perception_\tau, Action_\tau \rangle$ is defined by:

- Σ_τ: the set of states that agents of type τ can assume;
- $Perception_\tau : \Sigma_\tau \rightarrow W_F \times W_F$ function for agents of type τ: it associates each agent state to a pair (i.e. *receptiveness coefficient* and *sensitivity threshold*) for each field in F;
- $Action_\tau$: the behavioral specification for agents of type τ in terms of L*MASS language [20].

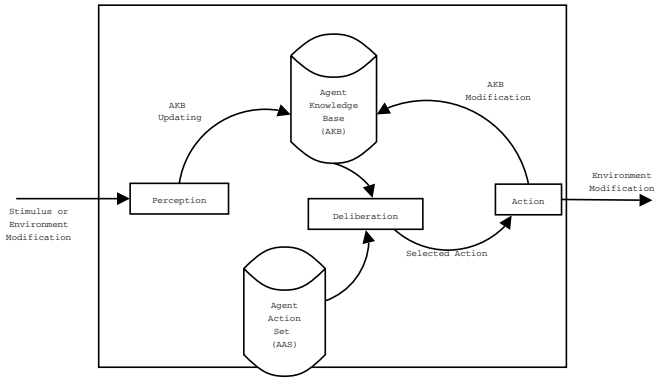

Fig. 1. A schematic representation of the internal architecture of SCA agents

Agent architecture (see figure 1) is composed by three tasks that define the agent actual behavior (i.e. *Perception*, *Deliberation*, and *Action*) and two knowledge containers:

- ***Agent Knowledge Base (AKB)*** is the internal representation of agent state and of its local perceptions (e.g. set of fields active in its site, set of empty sites in its surrounding). The AKB updating can be the effect of agent actions or of a change in the agent environment perceived by the agent (e.g. an adjacent site becomes empty, a new field reaches the agent site or the agent moves to another site).
- ***Agent Action Set (AAS)*** collects the set of actions that are allowed to the agent in terms of L*MASS language. AAS is defined according to the agent type and cannot change during agent execution. In the model presented in Section 4, we will experiment GP to evolve AAS towards an optimal strategy for evacuation.

SCA approach does not specify a standard way to define agents' perception, deliberation and action. SCA platform (the execution environment for SCA-based models) has been designed in order to be incrementally extended to several execution strategies. In our experiments we adopted a synchronous–parallel execution method for the system (i.e. at each timestep each agent update their AKB perceiving their local environment and selects the action to be performed). The phase between perception and execution is *deliberation* that is, the component of an agent responsible of conflict resolution between actions, when multiple actions are possible.

3 Genetic Programming

Genetic Programming (GP) [14] is a Machine Learning method that belongs to the family of Evolutionary Algorithms [15,16,17]. Its peculiar characteristic is that potential solutions (often called individuals) to be evolved are not fixed length strings of characters, as for Genetic Algorithms (GAs) or other evolutionary methods, but, generally speaking, *computer programs*. The fitness of a program is usually calculated by running it one or more times with a variety of inputs and seeing how close the program outputs are to a desired target. Programs can be represented as trees, lines of code, expressions in prefix or postfix notations, strings of variable length, etc. For tree-based GP, which is the original [14] and more popular version of GP and the one we use in this paper, the set of all the possible structures that can be generated is the set of all the possible trees that can be built recursively from a set of function symbols $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$ (used to label internal tree nodes) and a set of terminal symbols $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$ (used to label tree leaves). Each function in the fuction set $\mathcal{F}$ takes a fixed number of arguments, specifying its *arity*. Functions may include arithmetic operations ($+$, $-$, $*$, etc.), mathematical functions (such as `sin`, `cos`, `log`, `exp`), boolean operations (such as `AND`, `OR`, `NOT`), conditional operations (such as `If-Then-Else`), iterative operations (such as `While-Do`) and other domain-specific functions that may be defined. Each terminal is typically either a variable or a constant, defined on the problem domain. In the work presented in this paper the generated structure describes the behavior of a pedestrian agent. The terminal actions are atomic movements as: move towards, turn

to the left or turn to the right; while functions define nodes with two and three childs (see a deep descripion in section 4). Once such a formal language to code programs has been defined, the GP paradigm breeds those programs to solve problems by executing the following steps:

1. Generate an initial population of computer programs (or individuals).
2. Iteratively perform the following steps until the termination criterion has been satisfied:
   (a) Execute each program in the population and assign it a fitness value according to how well it solves the problem.
   (b) Create a new population by iteratively applying the following operations:
      i. Probabilistically select a set of computer programs to be reproduced, on the basis of their fitness (*selection*).
      ii. Copy some of the selected individuals, without modifying them, into the new population (*reproduction*).
      iii. Create new computer programs by genetically recombining randomly chosen parts of two selected individuals (*crossover*).
      iv. Create new computer programs by substituting randomly chosen parts of some selected individuals with new randomly generated ones (*mutation*).
3. The best computer program appeared in any generation is designated as the result of the GP process at that generation. This result may be a solution (or an approximate solution) to the problem.

Most commonly used termination criteria are: (1) at least one individual in the current population has a satisfactory fitness value, or (2) a prefixed number of generations has been executed.

4 Modeling Pedestrian Behavior for Evacuation with GP

In this work, we present a GP configuration inspired by the Artificial Ant on the Santa Fe trail benchmark to simulate the trajectory of a set of agents in a square space, with the aim of automaticallty generating suitable pedestrian paths for evacuation. The Artificial Ant on the Santa Fe trail is a problem where an artificial ant is placed on a regular toroidal grid, where some of the cells contain food pellets. The problem goal is to find a navigation strategy for the ant that maximizes its food intake. The ant starts in the upper left cell of the grid, identified by the coordinates (0, 0), facing east. It has a very limited view of its world. In particular, it has a sensor that can see only a single immediately adjacent cell in the direction the ant is currently facing.

Inspirating to this problem we defined a pool of N pedestrian agents, with $N > 1$ placed in a regular SCA space with Von Neumann neighborhood. The space is defined as a regular not toroidal square grid representing a room to be evacuated. In this scenario agents have a random starting postition and they have a direction that points to one of the adjacent sites; they have no knowledge

of the space (they can perceive only the adjacent place in the direction the agent is currently facing) and also the initial direction is randomly setup for each agent. No food is contained in the grid, and the only special place on the space is the given location identified as the exit.

Agent's behavior is built by GP and the set of possible structure that can be generated is the set of all possible trees that can be generated from $\mathcal{T} = \{Right, Left, Move\}$ and $\mathcal{F} = \{Progn2, Progn3\}$. The set of terminals $\mathcal{T}$ corresponds to the actions the agent can perform: turn right by 90°, turn left by 90° and move forward in the currently facing direction. In the set $\mathcal{F}$ function $Progn2$ takes two arguments and causes the agent to unconditionally execute the first argument followed by the second one, while $Progn3$ is analogous, but it takes three arguments, that are executed in an ordered sequence. An individual built with these sets $\mathcal{F}$ and $\mathcal{T}$ can be considered as a "*navigation program*" that allows the agent to navigate the grid. When the number of agents N is $N > 1$ the set of $\mathcal{F}$ and $\mathcal{T}$ builted by GP is applied to all the agents.

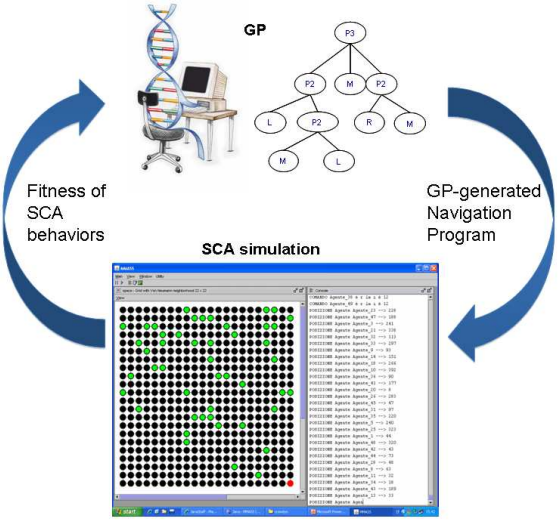

Fig. 2. The fitness of all Navigation Programs (the tree in the upper side of the figure) generated by the GP are obtained through a SCA simulation (a screenshot is shown in the lower side of the figure)

As fitness function, the number of agents which reach the exit and the total number of steps that they perform in order to reach the exit are considered (any action is considered as taking one time step). This turns the problem into a minimization one, with the globally optimal solution having a fitness value equal to 0. This problem specification represents a methodological example that can be very useful for SCA–based model calibration when data on real crowded situations have been collected. In these cases a suitable fitness function have to be specificated properly.

SCA simulation are used to obtain the fitness value of each individual behaving according to the *navigation program* (see figure 2). The experiments are performed in the synchronous mode that SCA organizes into two tasks: *deliberation* where each agent selects the next position according to its perception of the local environment, and *action* where agents try to perform what they have deliberated. All agents complete the phase *deliberation* before the *action* phase starts, and the order in which agents perform these phases are random at each simulation cycle. Conflics between the agents are managed by the SCA platform, and during the execution phase on the SCA–based simulation, if an agent cannot perform the action expressed by its behavior (i.e. the *navigation program*), it changes randomly its direction turning to the right or to the left (in figure 2 a screenshot of a SCA simulation). The possible causes that do not allow an agent to move according to its *navigation program* are the presence of obstacles: static obstacles are the borders of the space, or other structure that can be added to the space map, and dynamic obstacles are the other agents.

Agent behavior is limited to 600 time steps. This time-out limit is sufficiently small to prevent a random walk of the agent to cover all the 1024 squares (in case of a 32×32 space) before timing out.

5 Conclusions and Future Works

This paper presents an application of GP and CA-based approach to pedestrian dynamics. An analysis of the presented model is still ongoing. Future experiments on more structured spaces as buildings with a given number of rooms are in progress too. SCA approach allows specifying and simulating heterogeneous systems of agents with different behavioral rules. Therefore more complex studies on pedestrian dynamics considering different types of behaviors will be experimented and compared.

Other examples of current ongoing works concerns the study of aggregation phenomenon in *Open Crowds* [22] and, a SCA-based specification of *Affectons* formal framework proposed to study complex crowds' dynamics emerging from emotional interaction [23] and its adoption to study *emotional crowds* (e.g. at concerts or sports events). A further application for CA and GP in pedestrian dynamics context will concern the position optimization of obstacle and mobile structures in public spaces' design (see [1,24]).

References

1. Klupfel, H.: A Cellular Automaton Model for Crowd Movement and Egress Simulation. PhD thesis, Universität Duisburg-Essen (2003), `http://www.ub.uni-duisburg.de/ETD-db/theses/`
2. Dijkstra, J., Timmermans, H., Jessurun, A.: A multi-agent cellular automata system for visualising simulated pedestrian activity. In: Theoretical and Practical Issues on Cellular Automata. LNCS, pp. 29–36. Springer, Heidelberg (2001)

3. Schadschneider, A., Kirchner, A., Nishinari, K.: CA approach to collective phenomena in pedestrian dynamics. In: Bandini, S., Chopard, B., Tomassini, M. (eds.) ACRI 2002. LNCS, vol. 2493, pp. 239–248. Springer, Heidelberg (2002)
4. Proceedings of 4th Conference on Pedestrian Dynamics, Wuppertal, February 2008. Springer, Heidelberg (forthcoming, 2008)
5. Bandini, S., Federici, M.L., Manzoni, S., Vizzari, G.: Towards a methodology for situated cellular agent based crowd simulations. In: Dikenelli, O., Gleizes, M.-P., Ricci, A. (eds.) ESAW 2005. LNCS (LNAI), vol. 3963. Springer, Heidelberg (2006)
6. Still, G.K.: Crowd Dynamics. PhD thesis, University of Warwick, Warwick (2000), http://www.crowddynamics.com/
7. Bandini, S., Manzoni, S., Simone, C.: Enhancing cellular spaces by multilayered multi agent situated systems. In: Bandini, S., Chopard, B., Tomassini, M. (eds.) ACRI 2002. LNCS, vol. 2493, pp. 156–167. Springer, Heidelberg (2002)
8. Ferber, J.: Multi-Agent Systems. Addison-Wesley, Harlow (1999)
9. Bandini, S., Manzoni, S., Simone, C.: Modelling heterogeneity in multi agent systems. In: Shafazand, H., Tjoa, A.M. (eds.) EurAsia-ICT 2002. LNCS, vol. 2510, pp. 685–692. Springer, Heidelberg (2002)
10. Bandini, S., Manzoni, S., Simone, C.: Heterogeneous agents situated in heterogeneous spaces. Applied Artificial Intelligence 16, 831–852 (2002)
11. Schadschneider, A.: Cellular Automaton Approach to Pedestrian Dynamics - Theory. In: Pedestrian Dynamics, pp. 75–86. Springer, Heidelberg (2002)
12. Schadschneider, A.: Cellular Automaton Approach to Pedestrian Dynamics - Applications. In: Pedestrian Dynamics Applications, pp. 87–98. Springer, Heidelberg (2002)
13. Bandini, S., Federici, M.L., Manzoni, S., Redaelli, S.: A SCA–based model for Open Crowd aggregation. In: Pedestrian Dynamics. Springer, Heidelberg (2008)
14. Koza, J.: Genetic Programming. MIT Press, Cambridge (1992)
15. Holland, J.: Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor (1975)
16. Fogel, L., Owens, A., Walsh, M.: Artificial Intelligence Through Simulated Evolution. John Wiley and Sons, New York (1966)
17. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading (1989)
18. Kitazawa, K., Batty, M.: Pedestrian behaviour modelling: An application to retail movements using a genetic algorithm. In: 7th International Conference on Design and Decision Support Systems in Architecture and Urban Planning (2004)
19. Bandini, S., Manzoni, S., Vizzari, G.: Towards a platform for mmass–based simulations: focusing on field diffusion. Applied Artificial Intelligence 20, 327–351 (2006)
20. Bandini, S., Manzoni, S., Pavesi, G., Simone, C.: L*MASS: A language for situated multi-agent systems. In: Esposito, F. (ed.) AI*IA 2001. LNCS (LNAI), vol. 2175, pp. 249–254. Springer, Heidelberg (2001)
21. Schreckenberg, M., Sharma, S.: Pedestrian and Evacuation Dynamics. Springer, Berlin (2002)
22. Canetti, E.: Crowds and Power. The Noonday Press/Farrar, Straus and Giroux (1984)
23. Adamatzky, A.: Dynamics of Crowd–Minds. World Scientific, Singapore (2005)
24. Kretz, T., Schreckenberg, M.: The f.a.s.t.-model. In: Yacoubi, S.E., Chopard, B., Bandini, S. (eds.). LNCS, pp. 712–715. Springer, Heidelberg (2006)