Automatic Rule Identification for Agent-Based Crowd Models Through Gene Expression Programming

Jinghui Zhong[1], Linbo Luo[1], Wentong Cai[1], Michael Lees[2]
[1]School of Computer Engineering, Nanyang Technological University, Singapore
[2]Section Computational Science, University of Amsterdam, Netherlands
{jinghuizhong, lbluo, aswtcai}@ntu.edu.sg, m.h.lees@uva.nl

ABSTRACT

Agent-based modelling of human crowds has now become an important and active research field, with a wide range of applications such as military training, evacuation analysis and digital game. One of the significant and challenging tasks in agent-based crowd modelling is the design of decision rules for agents, so as to reproduce desired emergent phenomena behaviors. The common approach in agent-based crowd modelling is to design decision rules empirically based on model developer's experiences and domain specific knowledge. In this paper, an evolutionary framework is proposed to automatically extract decision rules for agent-based crowd models, so as to reproduce an objective crowd behavior. To automate the rule extraction process, the problem of finding optimal decision rules from objective crowd behaviors is formulated as a symbolic regression problem. An evolutionary framework based on gene expression programming is developed to solve the problem. The proposed algorithm is tested using crowd evacuation simulations in three scenarios with differing complexity. Our results demonstrate the feasibility of the approach and shows that our algorithm is able to find decision rules for agents, which in turn can generate the prescribed macro-scale dynamics.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence, Intelligent agents; I.6.3 [**Simulation and Modeling**]: Applications

General Terms

Algorithms

Keywords

Agent-Based Modelling, Crowd Simulation, Decision Rules, Evolutionary Algorithm, Gene Expression Programming

1. INTRODUCTION

Crowd modelling and simulation has now become an important and active research field with various applications including military training, evacuation analysis and digital game [5, 28, 31]. One of the common approaches to simulating the dynamics of a crowd is the agent-based modelling (ABM) paradigm. It models each individual in a crowd as an intelligent and autonomous agent. Each agent can perceive information and make decisions independently based on decision rules. Due to the significant increase in computing power, ABM has gained tremendous attention recently. Various agent-based crowd models [11, 20, 22, 24, 30] have been proposed to simulate the dynamics of crowds. A thorough survey on crowd modelling and simulation technologies can be found in [33].

However, designing a suitable agent-based model to reproduce particular crowd behaviors (e.g., crowd behaviors extracted from videos) is a challenging task. One of the key difficulties lies in designing suitable decision rules for agents. In addition, after decision rules are initially designed, model calibration is used to try and match the model dynamics with some observed real world dynamics. However, depending on the rules which have been defined, this may or may not be possible. Moreover, the process of calibration is also time consuming and tedious.

In this paper, instead of treating model design and calibration as a two-stage process, we propose an evolutionary framework, which is capable of automatically evolving decision rules and calibrating these rules. The proposed framework provides a feasible approach to extract the individual agent rules from a prescribed objective crowd behavior (e.g., those extracted from videos), so that an agent-based model using the rules can reproduce (or approximate) the objective crowd behaviors. This work is not only limited to crowd modelling, but also addresses a very fundamental question in ABM in general. That is, given some observed macroscopic emergent phenomena (e.g., crowd behavior), can we automatically identify a set of micro-scale dynamics (agent rules) which generate the macro-scale dynamics. We show that for some cases this type of process is feasible. Clearly, once the rules for reproducing specific desired emergent phenomena are obtained, we can adjust the rules to do sensitivity or "what-if" scenario analysis.

To automate the rule identification process, we first define our rule representation according to the general "sense-think-act" paradigm of human decision making process [12]. The problem of finding decision rules for particular crowd dynamics is then formulated as a symbolic regression problem. An evolutionary framework is then developed to solve the problem. In the proposed evolutionary framework, a group of candidate decision rules are randomly generated, each represented by a compact and linear gene expression

Appears in: *Alessio Lomuscio, Paul Scerri, Ana Bazzan, and Michael Huhns (eds.), Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), May 5-9, 2014, Paris, France.*

chromosome. A similarity measure mechanism is designed to compare the simulated results with the desired crowd behaviors and evaluate the fitness values of decision rules. The chromosomes are then evolved repeatedly using genetic operators, until meeting the terminal condition. To validate the effectiveness of the proposed algorithm, we use three scenarios of crowd egress simulation.

The remainder of the paper is organized as follows. Section 2 describes the related work. Section 3 describes the proposed evolutionary framework for identifying decision rules. Section 4 presents the simulation studies and experimental results. Finally, Section 5 describes the conclusions.

2. RELATED WORK

Design of decision rules is a crucial step in the agent-based crowd modelling process. The popular and effective method is to incorporate social and psychological factors to mimic the human decision process [10, 19, 21, 29, 30]. For example, Luo et al. [21] proposed an agent-based crowd model, which incorporates physiological, emotional and social attributes in the decision making process of agents. Fridman et al. [10], study the impacts of cultural attributes (e.g., personal spaces, speed, pedestrian avoidance side and group formations) on crowd dynamics. Integrating various factors to build effective decision rules for agent model is a difficult task which requires extensive experience and domain-specific knowledge.

Recently, evolutionary algorithms (EAs) have been utilized to assist the development and calibration of agent-based models. Calvez and Hutzler [1, 2] proposed to use genetic algorithm (GA) to tune parameters of an ant foraging model, and discussed some relevant issues such as the noise in fitness evaluation and parallel implementation. Stonedahl and Uri Wilensky [27] proposed to formulate model exploration task as search problems by designing appropriate objective functions. They introduced a new software tool named "Behavior Search" which uses GA to explore parameter space of agent-based model. In [26], Stonedahl and Uri Wilensky applied GA to calibrate the parameters of an Artificial Anasazi model. Junges and Klügl [17] proposed to use learning-based methodologies to assist the design of a multiagent model. They used a simple pedestrian evacuation scenario to show that learning techniques such as genetic programming have potential to assist the model development process. Smith [25] used GA to find specific behavioral rules for a Cowbird model, so as to generate certain static self-organized pattern such as the number of neighbors for individual agents.

As for agent-based crowd models, most of the existing work focuses on calibrating the parameter settings. For example, Decraene et al. [4] proposed an EA-based framework named CASE to explore the parameter settings for agent-based crowd models. Examples of agent-based crowd model calibration using CASE framework can be found in [3, 15]. Johansson [16] proposed an EA to adjust the parameters of the social force model so that the simulated crowd can match the crowd behavior extracted from video.

In general, our work differs from the above works in that we focus on evolving the structures (i.e., decision rules) of agent models for generating highly dynamic crowd behaviors. Moreover, we aim to identify rules from specific objective crowd behaviors over a period of time, so as to reproduce the macro-scale emergent behaviors. Our work provides an integrated framework to automate the rule design and calibration process for agent-based crowd models.

3. THE PROPOSED FRAMEWORK

In this section, we propose to use gene expression programming (GEP) to extract rules for agent-based models, aiming to reproduce some prescribed crowd behaviors. First, the problem of identifying decision rules from an objective crowd behavior is formulated as a symbolic regression problem. Then an evolutionary framework based on gene expression programming is developed to solve the problem.

3.1 Problem Definition

In agent-based crowd modelling, individuals are modelled as intelligent agents which have their own attributes and can make decision independently based on those decision rules. Modelling the decision making process of human is important for designing the decision rules. We assume that the agents follow the "sense-think-act" paradigm [12] to make decisions. The decision rules identified by our framework describe the "think" part of this process, or more generally the decision making process. For this work we simplify the think process to consider a single *choice*, whereby an agent has to make a single decision by considering a series of *options*. We therefore break down the decision making process further into a sensing phase, an assessment phase and a decision phase.

Sensing phase: In this phase, the decision maker perceives environment features $\{\lambda_1, \lambda_2, ...\}$ and discovers the alternative options $\{A_1, A_2, ...\}$. For example, when people evacuating from a room, the perceived information can be the distance to each exit gate and the flow rate of each exit gate. Different exit gates are alternative options for the same decision (i.e., which exit to choose).

Assessment phase: The second phase is to assess each alternation option. This phase is the most important and complex part in the decision making process. Whether an alternative option is good or not is dependent on the specific context, and various physical, social and psychological factors. We can regard the process of assessing an alternative option A_i as a function which maps perceived features to a reward value:

$$\delta_i = \phi(\lambda_{i,1}, \lambda_{i,2}, ..., \lambda_{i,n}) \qquad (1)$$

where ϕ is the reward function, $\{\lambda_{i,1}, \lambda_{i,2}, ..., \lambda_{i,n}\}$ is the feature values corresponding to A_i, n is the number of features considered, and δ_i is the reward value of A_i.

Decision phase: Given the above formulation, this phase simply chooses the best option, i.e. the option with the largest reward.

Based on the above assumptions of the decision making process, designing decision rules for agent-based crowd model is actually a process of finding the corresponding reward functions. In this paper, we focus on single decision cases, where agent only needs to make a single decision with multiple alternative options. The behavior of agents is determined by a single reward function (ϕ). However, in section 5 we discuss how to generalize the approach to multiple decisions.

Generally, the reward function ϕ can be modelled as a combination of perceived features and some functions (e.g., numerical operators, logical operators, or user defined oper-

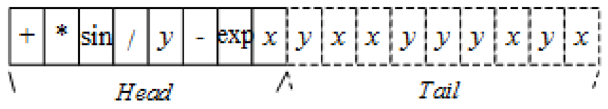


Figure 1: The structure of chromosome in gene expression programming.

ators). Denoted the feature set as:

$$T = \{\lambda_1, \lambda_2, ..., \lambda_n\} \tag{2}$$

where each λ_i is a perceived feature and the function set as:

$$F = \{f_1, f_2, ..., f_m\} \tag{3}$$

where each f_i is a function that maps one or more perceived features to a numerical value. If the desired crowd behavior (e.g., those extracted from video data) and the components of feature set and function set are given, finding the optimal decision rules for the model to reproduce the desired crowd behavior can be regarded as the following symbolic regression problem: *Given an objective crowd behavior, a feature set T, and a function set F, finding a symbolic reward function $\phi : \Re^n \to \Re$ to assess the alternative options of agents, so that the simulated crowd behavior based on the "sense-think-act" paradigm can match the objective behavior.*

Take a simple evacuation model for example, where people in the crowd choose from a set of exit gates (EGs) according to the following decision rules: *"Let E be the set of gates with distance (d) $< d_0$. If $E \neq \emptyset$ then choose the gate in E with the shortest distance, otherwise choose the gate with the largest width (w)."* In this model, the feature set can be

$$T = \{d, w\} \tag{4}$$

The function set can be

$$F = \{+, -, *, /, sig(x)\} \tag{5}$$

where $sig(x)$ is the sign function defined as:

$$sig(x) = \begin{cases} 0, \text{if } x \le 0 \\ 1, \text{otherwise} \end{cases} \tag{6}$$

A feasible symbolic reward function to reproduce the objective crowd behavior can be defined as:

$$\phi(d, w) = sig(d_0 - d)/d - sig(d - d_0)/w \tag{7}$$

3.2 Algorithm Design

Genetic programming (GP) is a well-known EA for identifying knowledge from large data set [6, 23]. Gene expression programming (GEP) is a specialization of GP which has been proposed by Ferreira [7]. It uses fixed length, linear strings of chromosomes to represent tree structure solutions and uses genetic operators to evolve the chromosomes. An advantage of GEP over traditional GPs is that its chromosomes are linear, compact and easy to genetically manipulate. The GEP is shown to perform better than traditional GPs on various applications [8] [32]. Therefore, in this paper, we adopt the GEP to search rules for the agent-based model. In traditional GEP, a chromosome may consist of one or more genes of equal length. In this paper, we consider a simplified GEP with single-gene chromosomes.

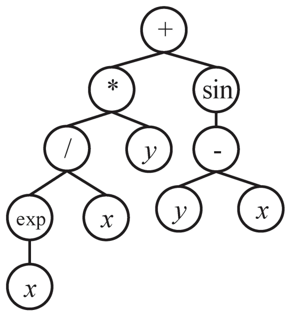


Figure 2: Example of an expression tree.

3.2.1 Chromosome Representation

In the proposed algorithm, each chromosome is represented by a vector of symbols with fixed length. The vector consists of two parts, namely the *"Head"* and the *"Tail"*, as shown in Fig. 1. Each element of the *"Head"* can be a function or a terminal, while each element of the *"Tail"* can only be a terminal. In this paper, the elements in the feature set are used as terminals. For example, given a function set $F = \{+, -, *, /, sin, exp\}$ and a feature set $T = \{x, y\}$, a typical GEP chromosome with length of 17 can be:

$$X = [+, *, sin, /, y, -, exp, x, y, x, x, y, y, y, x, y, x] \tag{8}$$

Each chromosome can be converted to an expression tree (ET) using a breadth first traversal scheme. For example, the chromosome expressed in (8) can be converted to an ET as shown in Fig. 2, which can be further expressed in a mathematical formula as

$$(exp(x)/x) * y + sin(y - x) \tag{9}$$

The length of the "Head" (H) and that of the "Tail" (L) are fixed. In order to ensure that any chromosome can result in a valid ET, H and L should have the following relation:

$$L = H \cdot (u - 1) + 1 \tag{10}$$

where u is the number of arguments of the function with the most arguments. In this paper, we consider $u \leqslant 2$. Thus, we have $L = H + 1$. Note that there may exist some redundant elements which are not useful for building the ET. But these redundant elements may become useful to build ETs in future evolution.

3.2.2 Fitness Evaluation

To evaluate the fitness of a chromosome, we need to define a distance measure between the simulated and prescribed crowd behaviors. By considering the spatial and temporal features of the two crowds, we can measure the distance of two crowds M_0, M_1 by

$$D(M_0, M_1) = \frac{\sum_{t=1}^{K} \left(\sum_{i=1}^{N} (|\xi_{i0}^t - \xi_{i1}^t|) \right)}{K \cdot N} \tag{11}$$

where K is the total number of time steps; N is the total number of representative points; ξ_{i0}^t is the behavior feature of M_0 at the i^{th} representative point of time step t ; and ξ_{i1}^t is the behavior feature of M_1 at the i^{th} representative point of time step t. We can choose N points that are evenly distributed in the region as the representative points to estimate the behavior feature of the crowd at each time step. There are various features that can be used, such as the flow rate and the density distribution. In this paper, we use the density distribution as behavior feature to measure the distance between two crowd behaviors. As suggested in Helbing's work [14], we use the following equation to compute the density of the crowd at location x at time step t:

$$\rho(x,t) = \sum_{i=1}^{C} \frac{1}{\pi R^2} exp[-\|r_i(t) - x\|^2/R^2] \qquad (12)$$

where C is the number of agents in the observed region; $r_i(t)$ is the position of the i-th agent and R is a constant (e.g., $R = 2$).

3.2.3 *Algorithm Operators*

The first step of the algorithm is to generate a set of random chromosomes (i.e., the initial population). Then in the second step, the fitness of each chromosome is evaluated by running the simulation using the reward function represented by the chromosome. The best chromosome of the generation is stored as the best-so-far solution. After that, the algorithm repeatedly carries out the following steps until meeting the terminal condition.

Selection: The selection operation selects promising chromosomes in the current population to form a new population for the next generation. In the proposed algorithm, the commonly used binary tournament selection strategy is used to select chromosomes.

Mutation: In this step, every element at any position of each chromosome is subject to a random change with a predefined mutation rate (pm). Note that elements of the chromosome can only be changed to a feasible random value according to their types (e.g., elements of *"Tails"* can only be a terminal).

Transposition: The transposition is performed on each chromosome with a probability of 0.1. It replaces some consecutive elements of the *"Head"* with an insertion sequence (IS). Firstly, a random insertion point in the "Head" part is chosen (except the starting point of the "Head"). The length of the IS can not be larger than that of the sequence from the insertion point to the end of the "Head". After that, the sequence downstream from the insertion point would be replaced by the IS.

Crossover: The crossover operator exchanges elements between two randomly chosen individuals with a predefined crossover rate (px). In this paper, two kinds of crossover are used: one-point, and two-point crossover operations. Both crossover operations work in the same manner as their GA counterparts [7].

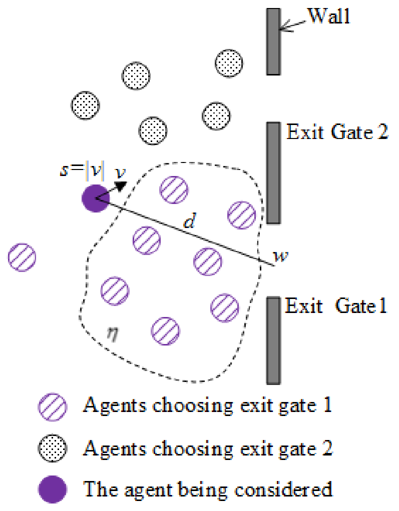
Fitness Evaluation: After generating the new population, the fitness values of all new individuals are evaluated and the best-so-far solution is updated.

4. SIMULATION STUDIES

In this section, we investigate the performance of the proposed algorithm. First, a crowd evacuation simulation for testing is desribed. We then describe three scenarios and two underlying behavioural rules used to test the algorithm (resulting in six case studies). Finally, we present the results and analysis.

4.1 A Crowd Evacuation Model for Testing

In this subsection, a crowd evacuation model is developed to test the algorithm. In the model, a number of agents are randomly distributed in an enclosed region. There are several exit gates (EGs) in the region. Agents are supposed to select an exit gate to move towards it and evacuate from the region.



Wall
Exit Gate 2
Exit Gate 1
s=|v|
v
d
w
η
Agents choosing exit gate 1
Agents choosing exit gate 2
The agent being considered

Figure 3: Four important factors to influence the action of agents.

A two-level model is adopted. At the lower level, we use the social force model [13] to guide the motions of agents. The social-force model uses virtual forces to guide the motions of agents. The force imposed on an agent is expressed as:

$$f_i = m_i \frac{dv_i}{dt} = f_{i_0} + \sum_{j(j \neq i)} f_{ij} + \sum_{w} f_{iw} \qquad (13)$$

where m_i and $\frac{dv_i}{dt}$ are the mass and acceleration rate of the agent; f_{i_o}, f_{ij} and f_{iw} are attractive force towards the goal, repulsive force from other agent and repulsive force from the static obstacles (e.g., wall) respectively.

On top of the social-force model, the higher-level decision making of the agents (i.e., choosing an exit) is determined by the decision rules. The final decisions are influenced by various factors that the agents perceive from the environment. In this evacuation model, we only consider four important factors as shown in Fig. 3.

The first factor (d) is the distance to the EG. Intuitively agents are more likely to choose an EG with a shorter distance. The second factor (w) is the width of the EG. A wider EG would lead to a larger evacuation rate and a shorter evacuation time. The third factor (η) is the number of agents ahead of the decision maker, who are moving towards the same EG. This factor is related to the waiting time for evacuation. A larger η would result in a longer evacuation time. The fourth factor (s) is the speed of the agent.

Before running our algorithm, we need some objective behaviors. In principle the objective behaviors should be obtained by extracting from video data and then rules can be evolved that match the real world crowd. However, this doesn't lend itself to testing the methodology. Instead, we define some decision rules (from literature), and run sample simulations to obtain output crowd dynamics. And then we use the sample simulation output as the prescribed or desired behavior. What we hope is that the approach should

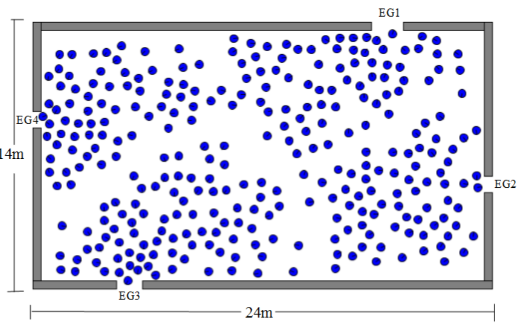


Figure 4: The first scenario.

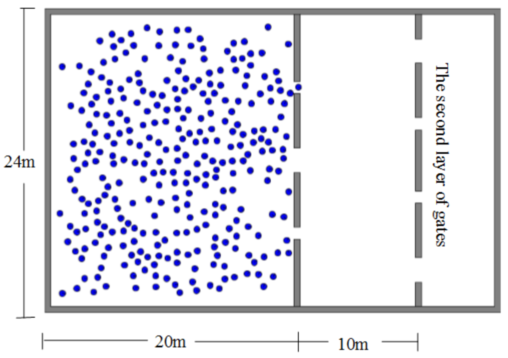


Figure 5: The second scenario.

be able to identify the same rules as in the sample simulations.

The decision rules are implicitly represented by a symbolic function: $\phi(\lambda_1, \lambda_2, ..., \lambda_n)$. Since we only consider four factors to determine the decision, the feature set of the reward function is:

$$T = \{d, \eta, s, w\} \tag{14}$$

The function set can be any numerical operators, logical operators, and user defined functions. In this study, we consider the following five basic numerical operators as the function set for simplicity:

$$F = \{+, -, *, /, f(x) = -x\} \tag{15}$$

4.2 Scenario Design

Based on the defined crowd evacuation model, we design three scenarios for testing. The first scenario is a 24m $\times$ 14m rectangle room with four walls, as shown in Fig. 4. Each wall contains an EG. The widths of the EGs are different. Agents reaching any of the four EGs are considered successfully evacuated from the room. The second scenario contains two layers of gates, as shown in Fig. 5. The crowd behavior in the second scenario is a bit more complicated than those in the first scenario, as the agents need to pass through the gates at the second layer. Agents which reach any of the second layer of gates are considered successfully evacuated from the room. The third scenario is designed based on a real world nightclub [18], as shown in Fig. 6. It contains 15 inner gates and five exit gates. An agent reaching any of the exit gates is considered successfully evacuated and removed from the simulation. The crowd behavior in the third scenario is more complicated than the first two scenarios, as there are more gates and rooms all of which are distributed irregularly. We design different scenarios with increasing complexity, so as to investigate the performance of our algorithm in identifying rules for different objective crowd behaviors.

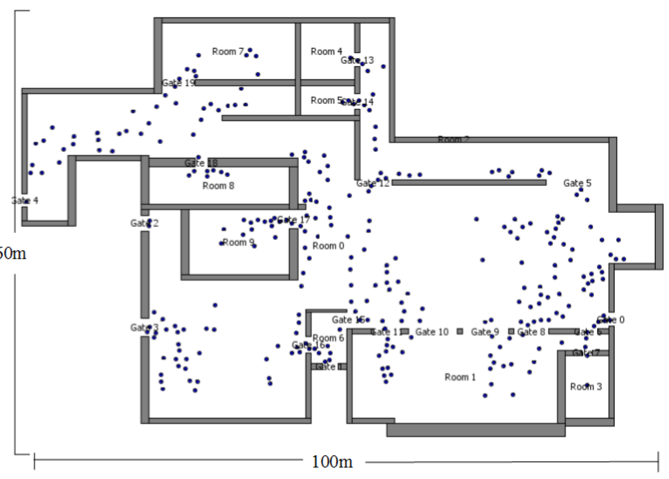


Figure 6: The third scenario.

4.3 Objective Crowd Behaviors

We generate six different objective crowd behaviors for testing. The six objective crowd behaviours are generated by running simulations on the three scenarios using two different evacuation strategies (underlying rules) respectively. The first evacuation strategy is named distance first ("DF"), where agents always choose the EG with the nearest distance. The second evacuation strategy is the LifeBelt method ("LB") [9]. This method recommends EG to agents based on three factors: the time to reach an exit gate (TEG), the number of individuals expected in the destination EG (EP), and the number of individuals that can possible escape through that exit per unit time (EC). The predicted evacuation time is calculated by:

$$\tau = TEG + \frac{EP}{EC} \tag{16}$$

Since EC is in proportion to the width of the EG (w), we simply use the following formula to calculate the predicted evacuation time.

$$\tau \approx \frac{d}{s} + \frac{\eta}{w} \tag{17}$$

Agents will choose the EG with the smallest τ. The LifeBelt method has been shown to produce good evacuation performance based on the empirical experiment with real humans [9].

4.4 Experiments and Simulation Results

In the simulations, we evenly divide the region into square grids and use the center points of the grids as the representative points for density estimation and then fitness evaluation. The lattice size of the 1^{st}, 2^{ed}, and 3^{rd} scenarios are set to be 1m, 1m, and 2m respectively. The preferred speed of all agents is set to be s = 1m/s and the parameters of the GEP algorithm are set as follows: population size ($popsize$) = 10, the length of "Head"(H) = 6, maximum generation ($maxgen$) = 100, mutation rate (pm) =

Table 1: Algorithm performances on the six instances

CASE	*Succ*	*Avg*	*Std*
S1DF	100%	0.0	0.0
S1LB	36.7%	0.0127	0.0115
S2DF	100%	0.0	0.0
S2LB	50%	0.0066	0.0077
S3DF	100%	0.0	0.0
S3LB	23.3%	9.9E-4	7.5E-4

Table 2: Examples of the best solutions found by the proposed algorithm.

CASE	Original ϕ	Simplified ϕ
S1DF	$-(d)$	$-d$
	$(s/((s+d)--(d)))$	$1/(1+2d)$
	$((s+-(w))+(w-d))$	$1-d$
S1LB	$(((s/d)-(\eta/w))+-(d))$	$1/d-\eta/w-d$
	$-((d+((\eta/s)/w)))$	$-(d+\eta/w)$
	$((\eta/-(w))-d)$	$-(d+\eta/w)$
S2DF	$(((\eta*s)-d)-n)$	$-d$
	$(((w*s)-w)-d)$	$-d$
	$(s/(s+(d*d)))$	$1/(1+d^2)$
S2LB	$(((d-d)-(\eta/w))-(s*d))$	$-(d+\eta/w)$
	$-((((s/s)+d)+(\eta/w)))$	$-(1+d+\eta/w)$
	$-((((w+\eta)/w)+(s+d)))$	$-(2+d+\eta/w)$
S3DF	$((s+-(d))*d)$	$1-d^2$
	(s/d)	$1/d$
	$((s-d)*d)$	$d-d^2$
S3LB	$(s/((s+d)+(\eta/w)))$	$1/(1+d+\eta/w)$
	$-((d+((\eta/w)+s)))$	$-(1+d+\eta/w)$
	$((-(\eta)/(w*s))+-(d))$	$-(d+\eta/w)$

0.1, and crossover rate $(px) = 0.7$. As evolutionary algorithms are stochastic search algorithms, which may generate different results in different runs, we run the proposed algorithm for 30 independent times on each test instance. The simulation results and related videos can be found at *http://crowds.sce.ntu.edu.sg/index.php/research/mrcm*.

Table 1 lists the experimental results on the six case studies that are created with three scenarios using both "DF" and "LF" strategy respectively. In Table 1, "SiDF" is the case where the objective crowd is obtained from scenario i and using the "DF" evacuation strategy, while "SiLB" is the case where the objective crowd is obtained from scenario i and using the "LB" evacuation strategy. "*Succ*" is the successful rate of finding the optimal solution (i.e., fitness $(D) = 0$); "*Avg*" is the average of the best fitness values found in the 30 runs, and "*Std*" is the standard deviation of the best fitness values found in the 30 runs. It can be observed that the proposed algorithm has 100% successful rate on all three test instances which use the "DF" evacuation strategy. As for the three instances which use the "LB" strategy, the proposed algorithm is also able to find optimal rules to reproduce the objective crowd behaviors in some runs (i.e., "*Succ*" > 0). Clearly, the successful rate of the three instances using "LB" strategy can be enhanced by increasing the computation budget (e.g., increase the size of population and the maximum number of generations).

Fig. 7 shows the average of the best-so-far fitness values (f_{best}) versus the number of fitness evaluations. To clearly

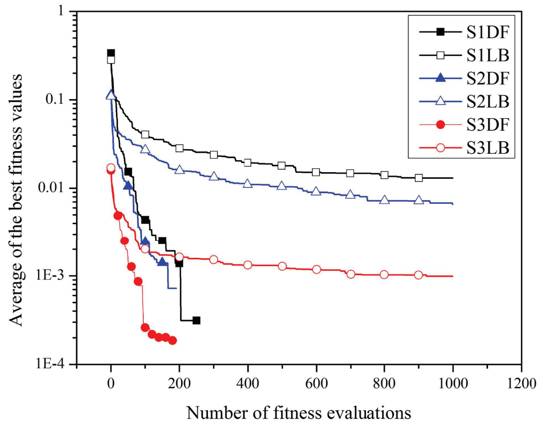


Figure 7: Average of the best fitness values versus the number of fitness evaluations.

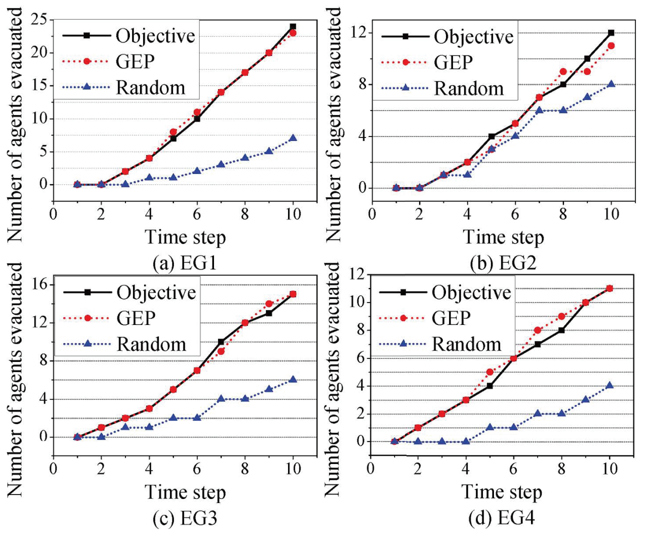


Figure 8: The accumulated number of agents evacuated from the gates.

show the evolution trend of f_{best}, we use a log plot. When f_{best} reduces to zero, the curve will stop. The results show that the f_{best} quickly drops to zero on the three instances which use the "DF" evacuation strategy. This validates the high efficiency of the proposed algorithm in identifying relatively simple decision rules. Meanwhile, as for the other three instances, the f_{best} becomes smaller as the number of evaluations increases. That means the reproduced crowd behaviors are becoming closer to the desired ones.

Table 2 shows 18 examples of the best solutions found by the proposed algorithm. The fitness values of all 18 solutions are equal to zero (i.e., $D = 0$). The results demonstrate that our algorithm is able to find different optimal rules to produce the same objective crowd behavior. For example, in the first three rows of Table 2, the three optimal solutions are $-(d)$, $(s/((s+d)--(d)))$, and $((s+-(w))+(w-d))$. They can be simplified as $-d$, $1/(1+2d)$, and $1-d$ (using $s = 1m/s$). Although they have different expressions, they all lead to the same decision, i.e., always choose the nearest EG.

Table 3: Impacts of the number of agents (C) on the performances of the algorithm

C	OS	$Succ$	Avg	C	OS	$Succ$	Avg
20	DF	100%	0.0	20	LB	36.7%	0.00122
50	DF	100%	0.0	50	LB	30%	0.0029
100	DF	100%	0.0	100	LB	20%	0.0081
150	DF	100%	0.0	150	LB	16.7%	0.0135
200	DF	100%	0.0	200	LB	36.7%	0.0127
500	DF	100%	0.0	500	LB	33.3%	0.0284

To visually analyze the solution found by the algorithm in the worst case we run a simulation using the worst best-so-far solution found by the GEP in the "S1LB" test instance. The worst best-so-far solution is $\phi = -((s * (-(w) + d)))$, with the fitness value being $D = 0.04441$. We also run a simulation where each agent choose a random EG. Fig. 8 shows the accumulated number of evacuated agents versus the time step at the four EGs. In Fig. 8, "Objective" represents the results of the objective crowd behavior, "GEP" represents the results of the simulation where agents choose EGs based on ϕ, and "Random" represents the results of the simulation where agents choose EGs randomly. It can be observed that the "random" results at four EGs are much different from the "Objective" results, while the "GEP" results are always very similar to the "Objecitve" results. These results demonstrate that the proposed algorithm can effectively discover rules for the agent-based crowd model to reproduce (or approximate) desired crowd behaviors.

4.5 Algorithm Analysis

In this subsection, we study the impacts of the number of agents (C) on the performance of the algorithm. We generate 12 test instances where the objective crowd behaviors are obtained from the first scenario. The first six instances use the "DF" evacuation strategy, while the last six instances use the "LB" evacuation strategy. For each evacuation strategy, the number of agents for the six test instances are set to be 20, 50, 100, 150, 200 and 500 respectively. Other parameters are set the same as in the previous simulations. Our proposed algorithm is carried out for 30 independent evolution runs on each test instance and the successful rate ("*Succ*") and the average of the 30 best-so-far fitness values ("*Avg*") are recorded for analysis.

Table 3 lists the "*Succ*" and the "*Avg*" versus the number of agents. From these results, we could infer that: 1) For objective crowd behaviors which use relatively simple rules such as the "DF" strategy, our method is very effective in finding the optimal decision rule, regardless of the number of agents. For example, our method achieves 100% successful rate on all six test instances where the objective crowd behaviors using "DF" evacuation strategy; 2) As for crowd behaviors that use more complex rules, our algorithm is still able to find the optimal rule in some cases, even when the number of agents increases from small to large (i.e., "*Succ*" > 0). The fluctuating values of "*Succ*" show that the number of agents dose not have significant impacts on the performance of the algorithm; and 3) The "Avg" generally increases as the number of agents increases. This is due to the fact that the density would increase when the number of agents increases.

5. CONCLUSIONS

In this paper, we have proposed an evolutionary framework to identify decision rules for agent-based crowd model so as to reproduce specific objective crowd behaviors. First of all, the problem of identifying optimal decision rules from objective crowd behaviors is formulated as a symbolic regression problem. Then an evolutionary framework based on the gene expression programming is developed to solve the problem. The proposed framework is tested through test instances with different levels of complexity and the results demonstrate that our framework is able to find decision rules to reproduce specific desired crowd behaviors.

This proposed framework automates the rule design and calibration process in agent-based modelling. The key issue in applying the proposed framework is to obtain the behavior features of objective crowd behaviors, such as the density distribution of agents over time. If these features are available (e.g., extracted from videos), then our framework can be used to find rules that generate the macro-scale dynamics.

There are several interesting future work directions. The first direction is to extend the proposed framework to construct and calibrate multi-decision crowd model, where agents can take multiple types of decisions. For example, in the evacuation scenario, an agent needs to not only choose an exit gate, but also search missing group members. In such cases, multiple reward functions are required to control the behaviors of all agents. By incorporating multi-gene techniques of the gene express programming, the proposed framework can be extended to evolve multi-decision crowd models. The second direction is to apply the proposed framework to optimize the decision rules, aiming to achieve certain desired objectives. For example, finding the best evacuation rules to minimize the evacuation time. The third promising direction is to extend the proposed framework to evolve both decision rules and parameters simultaneously. In many applications where the given model is incomplete or partially not well-defined, tuning both parameters and structures (i.e., decision rules) can achieve better performance.

6. REFERENCES

[1] B. Calvez and G. Hutzler. Parameter space exploration of agent-based models. In *Knowledge-Based Intelligent Information and Engineering Systems*, pages 633–639. Springer, 2005.

[2] B. Calvez and G. Hutzler. Automatic tuning of agent-based models using genetic algorithms. In *Multi-Agent-Based Simulation VI*, pages 41–57. Springer, 2006.

[3] J. Decraene, M. Chandramohan, F. Zeng, M. Y. H. Low, and W. Cai. Evolving agent-based model structures using variable-length genomes. In *Proceedings of the 4th International Workshop on Optimisation in Multi-Agent Systems at AAMAS*, volume 11, 2011.

[4] J. Decraene, M. Y. H. Low, F. Zeng, S. Zhou, and W. Cai. Automated modeling and analysis of agent-based simulations using the case framework. In *Proceedings of the 11th International Conference on Control Automation Robotics & Vision (ICARCV)*, pages 346–351, 2010.

[5] B. D. Eldridge and A. A. Maciejewski. Using genetic

algorithms to optimize social robot behavior for improved pedestrian flow. In *Proceedings of 2005 IEEE International Conference on Systems, Man and Cybernetics*, volume 1, pages 524–529, 2005.

[6] P. G. Espejo, S. Ventura, and F. Herrera. A survey on the application of genetic programming to classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 40(2):121–144, 2010.

[7] C. Ferreira. Gene expression programming: a new adaptive algorithm for solving problems. *arXiv preprint cs/0102027*, 2001.

[8] C. Ferreira. *Gene expression programming*. Springer Berlin, 2006.

[9] A. Ferscha and K. Zia. On the efficiency of lifebelt based crowd evacuation. In *Proceedings of the 2009 13th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications*, pages 13–20, 2009.

[10] N. Fridman, G. A. Kaminka, and A. Zilka. The impact of culture on crowd dynamics: an empirical approach. In *Proceedings of the 12th International Conference on Autonomous Agents and Multi-agent Systems*, pages 143–150, 2013.

[11] S. J. Guy, M. C. Lin, and D. Manocha. Modeling collision avoidance behavior for virtual humans. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 2-Volume 2*, pages 575–582, 2010.

[12] N. K. Hayles. Computing the human. *Theory, Culture & Society*, 22(1):131–151, 2005.

[13] D. Helbing, I. Farkas, and T. Vicsek. Simulating dynamical features of escape panic. *Nature*, 407(6803):487–490, 2000.

[14] D. Helbing, A. Johansson, and H. Z. Al-Abideen. Dynamics of crowd disasters: an empirical study. *Physical review E*, 75(4):046109, 2007.

[15] N. Hu, J. Decraene, and W. Cai. Effective crowd control through adaptive evolution of agent-based simulation models. In *Proceedings of the Winter Simulation Conference*, pages 213–224, 2012.

[16] A. Johansson, D. Helbing, and K. S. PRADYUMN. Specification of the social force pedestrian model by evolutionary adjustment to video tracking data. *Advances in Complex Systems*, 10(supp02):271–288, 2007.

[17] R. Junges and F. Klügl. Evolution for modeling: a genetic programming framework for sesam. In *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation*, pages 551–558. ACM, 2011.

[18] P. Kamkarian and H. Hexmoor. Crowd evacuation for indoor public spaces using coulomb's law. *Advances in Artificial Intelligence*, 2012:4, 2012.

[19] S. Kim, S. J. Guy, D. Manocha, and M. C. Lin. Interactive simulation of dynamic crowd behaviors using general adaptation syndrome theory. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, pages 55–62, 2012.

[20] L. Luo, S. Zhou, W. Cai, M. Lees, M. Low, and K. Sornum. HumDPM: A decision process model for modeling human-like behaviors in time-critical and uncertain situations. *Transactions on Computational Science XII*, pages 206–230, 2011.

[21] L. Luo, S. Zhou, W. Cai, M. Y. H. Low, F. Tian, Y. Wang, X. Xiao, and D. Chen. Agent-based human behavior modeling for crowd simulation. *Computer Animation and Virtual Worlds*, 19(3-4):271–281, 2008.

[22] S. R. Musse and D. Thalmann. Hierarchical model for real time simulation of virtual human crowds. *IEEE Transactions on Visualization and Computer Graphics*, 7(2):152–164, 2001.

[23] M. Schmidt and H. Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009.

[24] W. Shao and D. Terzopoulos. Autonomous pedestrians. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 19–28, 2005.

[25] V. A. Smith. Evolving an agent-based model to probe behavioral rules in flocks of cowbirds. In *ALIFE*, pages 561–568, 2008.

[26] F. Stonedahl and U. Wilensky. Evolutionary robustness checking in the artificial anasazi model. In *Proceedings of the AAAI fall symposium on complex adaptive systems: Resilience, robustness, and evolvability*, 2010.

[27] F. Stonedahl and U. Wilensky. Finding forms of flocking: Evolutionary search in abm parameter-spaces. In *Multi-Agent-Based Simulation XI*, pages 61–75. Springer, 2011.

[28] D. Thalmann. *Crowd simulation*. Wiley Online Library, 2007.

[29] M. C. Toyama, A. L. Bazzan, and R. Da Silva. An agent-based simulation of pedestrian dynamics: from lane formation to auditorium evacuation. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 108–110, 2006.

[30] J. Tsai, N. Fridman, E. Bowring, M. Brown, S. Epstein, G. Kaminka, S. Marsella, A. Ogden, I. Rika, A. Sheel, et al. Escapes: evacuation simulation with children, authorities, parents, emotions, and social comparison. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, pages 457–464, 2011.

[31] B. Ulicny and D. Thalmann. Crowd simulation for interactive virtual environments and vr training systems. In *Computer Animation and Simulation 2001*, pages 163–170. 2001.

[32] C. Zhou, W. Xiao, T. M. Tirpak, and P. C. Nelson. Evolving accurate and compact classification rules with gene expression programming. *IEEE Transactions on Evolutionary Computation*, 7(6):519–531, 2003.

[33] 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 Transactions on Modeling and Computer Simulation (TOMACS)*, 20(4):20, 2010.