2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology

A Normative Model for Behavioral Differentiation

Benoit Lacroix
Renault, Technocentre
Technical Center for Simulation
and LIFL, University of Lille
benoit.lacroix@renault.com

Philippe Mathieu
LIFL, University of Lille
Cite Scientifique Bat M3
59655 Villeneuve d'Ascq
philippe.mathieu@lifl.fr

Andras Kemeny
Renault, Technocentre
Technical Center for Simulation
78288 Guyancourt
andras.kemeny@renault.com

Abstract

In multi-agent systems simulations, reproducing realistic behaviors is a crucial issue. Their variety and consistency are important factors, usually not specifically considered. In this paper, we propose a behavioral differentiation model designed (1) to generate various and consistent behaviors, and (2) to control the determinism of this generation. Based on a normative system and a nondeterministic generation engine, it allows users adapting it easily to their various needs. Finally, we show its application to the traffic simulation software developed and used at Renault, SCANeR© II.

1 Introduction

A crucial issue in multi-agent systems simulations addresses the realism of agents' behaviors. Indeed, when working on simulations reproducing real world situations, unrealistic behaviors get users to question their results. Many elements influence the realism; two major criteria are the variety and the consistency of the behaviors. Without a large variety of behaviors, a high level of realism is unlikely to be observed. This variety is often obtained by providing the agents with individual characteristics, like clothing for agents in virtual crowds [8]. Behaviors have to be various, but also consistent. It is an essential criterion for the validity and the credibility of the simulations, which will be questioned if aberrant behaviors appear. Usually, no mechanism is provided to generate a broad variety of behaviors, while guarantying their consistency. The intrinsic properties of normative systems provide an interesting means to describe the behaviors of the agents [4], and offer the possibility to use deviations to increase the behavioral variety. Managing the determinism is the common point in the various identified needs of the users. The generation of the behaviors uses thus nondeterministic principles, like applied in displacement models [10]. It allows not only generating the behaviors, but also controlling the simulation itself.

In this paper, we present first the proposed behavioral differentiation model, before describing the data structure and the generation engine. Finally, we show an application to traffic simulation and discuss experimental results.

2 Description of the model

Many multi-agent systems take benefits from the characteristics of normative systems [1, 11]: regulation possibilities, coordination and cooperation improvements... However, in addition to these, they can be used as generic structures allowing describing all kind of elements, from the decision model of the simulation [2] to the specific skills of the agents [9]. This provides a high genericity to the system. Moreover, the normative models intrinsically handle the notion of violations, which can be exploited to generate variety. They are thus adapted structures to answer our specific needs [6]. A generation engine is also needed to produce the behaviors. It has to meet different requirements. First, be generic and flexible: it should be easily adaptable to various contexts, while usable by functional as well as technical experts. Second, keep under control the reproducibility of the simulations and the consistency of the behaviors. The management of the determinism is central in all these points, so the mechanism is based on the use of nondeterminism. It is generalized to be available at any level of the simulation, depending on the control users wish.

The data structure describes the behavior of the agents as norms. The selection of the parameters characterizing the behaviors within the norms limits guaranty their consistency, and two different means lead to behavioral variety. First, using the definition of the norms themselves: as the definition domains can be as wide as wished, a large potential of behaviors is available. Second, using the generation engine, during the instantiation of behaviors. In this last case, violations may eventually be allowed to experiment unexpected behaviors. However, the consistency criteria can then obviously not be guarantied any more.

The data model associated to the generation engine can

978-0-7695-3496-1/08 $25.00 © 2008 IEEE
DOI 10.1109/WIIAT.2008.287

be used as an external tool providing input parameters to the simulation. In this case, the simulation keeps using its internal decision model, and only requires the differentiation model for specific cases (for example parameters creation, like presented in section 5). Another possibility is to use the generation engine to manage the whole simulation, if the norms described in the data model represent a decision model.

3 A data structure based on norms

3.1 Semantic

A classical terminology is used to present the normative model. However, some definitions have been slightly adapted to our context, to allow an efficient description of the data structure.

Definition 1: Institution *We define an Institution as a tuple* $\langle P, D_P, \Gamma_P, P_i, P_e \rangle$ *where:* P *is a finite set of parameters ;* $D_P = \{d_p, \forall p \in P\}$ *is a set of definition domains, defined for each element in* P *;* $\Gamma_P = \{\gamma_p : d_p \rightarrow \mathbf{R}, \forall p \in P\}$ *is a set of distance functions, defined for each element in* P *;* P_i *is a set of institutional properties ;* P_e *is a set of environmental properties.*

The purpose of the institution is to manage the norms in the environment. It is mainly used as a set of parameters and definition domains, and provides a distance function for each of the definition domains. A set of institutional and environmental values keeps track of the contextual elements. Note that parameter is used here with a wide meaning: it can be an action rule associated to its pre-conditions.

Definition 2: Norm *We define a Norm as a tuple* $\langle I, P_n, D_{P_n}, P_{n_i}, P_{n_e} \rangle$ *where:* I *is the institution the norm refers to ;* $P_n \subset P$ *is the subset of parameters associated to the norm ;* $D_{P_n} \subset D_P$ *is the subset of definition domains ;* P_{n_i} *is a set of institutional properties ;* P_{n_e} *is a set of environmental properties.*

Norms are subsets of the institution parameters, associated to subsets of the definition domains. They handle specific sets of institutional and environmental properties, which can specialize institution's ones. Conflicting norms are allowed; their preference ordering and their interpretation are left to the decision model of the agents. Several norms can be defined for the same environment, and norms can have non-empty intersections.

Definition 3: Behavior *A Behavior is defined as a tuple* $\langle N, P_b, V_{P_b}, P_{b_i}, P_{b_e} \rangle$ *where :* N *is a reference to the instantiated norm ;* P_b *is a subset of the set of parameters defined in the instantiated norm ;* V_{P_b} *is the set of values associated to the parameters ;* P_{b_i} *is a set of institutional properties ;* P_{b_e} *is a set of environmental properties.*

A behavior describes the instantiation of a norm. Each element of the behavior is described by a parameter taken from the corresponding norm, and a value associated to this parameter. This value can be taken in or outside the definition domain associated to this parameter in the norm.

3.2 Quantification of the violations

In the proposed model, we take advantage of the behavioral variety offered by the possibility to violate the norms: a behavior having at least one of the values of its parameters outside the definition domain specified in the norm is in violation. Their creation happens during the instantiation process of the behaviors: values may be taken in or outside the definition domain, according to the generation engine used. These violations can be quantified using two criteria: first, using the number of parameters' values which are outside the limits; second, using the distance functions defined in the Institution. This quantification can be exploited to exclude too deviant behaviors, as we are able to measure the deviations. It can also be used to create unexpected behaviors and study their influence on the simulations.

For instance, consider the behavior of the drivers regarding the safety distance on roads. This behavior is part of an institutional environment (the Highway Code), using norms (the 2 seconds rule). The norm can be described different ways: for example, either the parameter $safetyTime$ has a definition domain $\{2\,s\}$, or $safetyTime$ corresponds to a normal distribution of mean $\mu = 1.5\,s$ and standard deviation $\sigma = 0,25$ (more realistic). Suppose a driver adopts a safety time $t_s = 1.2\,s$. With the first norm's definition, he is in violation ($1.2 \neq 2$). If we define the distance function as $\gamma = |2\,s - t_s| \,/\, 2\,s$, the deviation is 40 %. With the second definition, no violation is observed.

4 The generation engine

4.1 Description

The instantiation from norm to behavior required by the data structure is achieved using a generic generation engine, based on nondeterminism: it assigns objects to the agents, while managing the determinism's level of this attribution. Each agent is associated to a finite set O of available objects. All objects in O are balanced with a probabilistic factor p_o, and a deterministic process d is associated to the agent to select the next object. Let p be a random parameter, $p \in \ ]0, 1]$. At each time step t, $O_t \subset O$ is the set of objects which can be selected. Using the probability $1/p$, the agent uses randomly one of the objects in O_t, else it uses the deterministic process d to choose it. p itself is randomly chosen with a probability q. If $q = 0$, the probability to choose randomly an object is null, and the resulting behavior of the agent is purely deterministic. If $q = 1$, the object is randomly chosen at each step, and the behavior is

purely non-deterministic. In all other cases, the behavior is semi-deterministic. The engine runs in three steps. First, the parameter q is chosen by the user or loaded from the configuration. It can be modified on runtime, or remain fixed during the whole simulation. Second, p is computed at each time step, and used to select the next object o. Finally, the agent applies o according to its own probability p_o.

This generic generator can be applied at different levels in the simulation (at the level of the agents to select their skills, at the level of the simulation to select global parameters...), and allows following the provided deterministic process, or easily introducing and controlling nondeterminism.

4.2 Working with the data structure

This generation engine can be used to instantiate the behaviors of the data structure. The nondeterminism possibilities naturally introduce the desired behavioral differentiation. To do so, the generation engine is used at the norm level: $O = P_n$. $O_t = P_b \subset P_n$ is the set of available parameters for the processed Behavior. The model is used to instantiate the value v_{p_b} of the parameter from the corresponding definition domain d_{p_n} of the norm. The definition of each parameter $p_n \in P_n$ includes the probabilistic factor pf_{p_n}. The factor q is managed by the behavior, as it can differ for each agent. The probabilistic factor p is generated as $p = f(q)$. Finally, $v_{p_b} = g_{p_n}(d_{p_n}, p, pf_{p_n})$. Any data model able to be described by the formalism presented in section 3 can be instantiated with the generation engine. For instance:

Soccer player agents: consider only the behavior "shoot" where soccer player agents have to choose if they shoot right or left in the goal: we have a norm $shoot$, using only one parameter $direction$ of definition domain $\{left, right\}$. Suppose the deterministic process being the $\{right, left\}$ sequence repeated. If $q = 0$, the agent will always shoot right, left, and repeat the sequence. If $q = 1$, it will always shoot randomly right or left. (*http://www2.lifl.fr/SMAC/projects/cocoa/football.html*).

Market agents: agents emit trading desires to the market, which are interpreted according to market model trading rules. These desires are defined by a composition of three characteristics: a direction, a price and a quantity. The direction is the minimal requirement to get a valid desire (emitting a desire to a market without saying if one wants to buy or sell makes no sense). We have built four norms of agents: zero-intelligent traders, chartists, fundamentalists and speculators. Their characteristics can be generated with the model, using the method presented in section 5.

Table 1. Institution.

p_1 = maximal speed	$d_{p_1} = [0, +\infty]$
p_2 = safety time	$d_{p_2} = [0, +\infty]$
p_3 = overtaking risk	$d_{p_3} = [-1, 3]$
p_4 = speed limit risk	$d_{p_4} = [0, +\infty]$
p_5 = observe signs	$d_{p_5} = \{true, false\}$
p_6 = observe priority	$d_{p_6} = \{true, false\}$

5 Application to traffic simulation

5.1 Implementation

One of the applications of this work is to reproduce realistic behaviors in traffic simulation, to improve the immersion of human drivers in driving simulators. The model has been applied to the driving simulation software developed and used at Renault, SCANeR© II. In SCANeR© II, the autonomous vehicles use a classical perception-decision-action architecture as reasoning basis [7]. This decision model takes into account different pseudo-psychological parameters: the "maximal speed" (maximal acceptable speed for the driver), the "safety time" (related to the security distance), the "overtaking risk" (risk a driver will accept to overtake), the "speed limit risk" (to bypass speed limits), and finally "observe priority" and "observe signs" (boolean rules for the respect of signalization and priorities).

We chose in this work to apply the proposed differentiation model directly on the available pseudo-psychological parameters. They influence the resulting behaviors, and are adapted inputs to the traffic model. The description of the set of available parameters constitute the institution (table 1). P_i and P_e are empty sets, and Γ_P holds functions computing the distance to the mean of the intervals.

The experiment was done on a database representing a highway, on a 11 km long section. The vehicles were generated using a traffic demand of 3800 veh/h, and their data recorded using detectors at kilometer points 2.2, 6 and 10.8. After the initial creation by the differentiation model, the traffic model of the application handles all the vehicles. They were instantiated using three different sets of norms. In the first one, *no norms*, all the vehicles are created with the same parameters (the behavioral differentiation model is deactivated). In the second one, *normal driver only*, one norm is used, defining only one parameter, the maximal speed. The generation engine computes its value from a normal distribution of mean $\mu = 125$ and standard deviation $\sigma = 10$, truncated at 100 and 140 km/h. In the third one, *all norms*, three norms are used: cautious, normal and aggressive drivers. Each norm defines all the parameters, which definition domains are truncated normal distributions reflecting real world values.

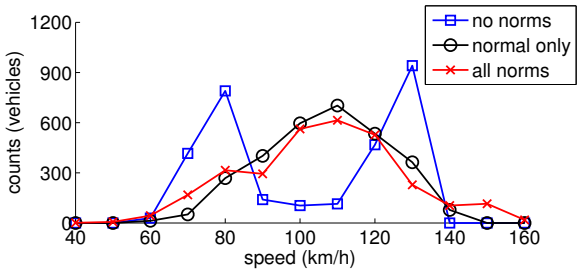

Figure 1. Speed of vehicles at km point 6

5.2 Experimental results and discussion

The figure 1 represents the distribution of the vehicles' speeds at the second detector. Without using any norm, the recorded speeds are either below 90 km/h (46 %), or around 130 km/h (40 %). In this case, the parameters of the vehicles are too similar to allow them adapting easily to small changes in traffic flows: the left lane remains slow, the right one fast, and vehicles can not pull in or over. Using one norm, the resulting distribution is more balanced, but the average speed remains relatively low (100.4 km/h). The last case, using three norms, presents a similar aspect, with a wider distribution of the speeds and a slightly increased average speed (103.7 km/h). While guarantying the consistency of the behaviors with the use of norms, an increased variety of behaviors is observed in the simulation.

Different elements have to be discussed. First, the norms and values were chosen according to usual classifications established by driving psychologists. However, the values have been fixed empirically, and calibration with real data is currently under work to improve this point. Second, various improvements on traffic dynamicity (overtakings number...) do not appear in statistical results, and new indicators have to be introduced to reflect them. Finally, we did not exploited in these simulations the possibility to generate violating behaviors. They will be introduced in further experiments, to simulate for instance drunk drivers.

6 Related works and conclusion

Some works have used norms in the context of traffic simulation, to enhance traffic control strategies [2] or to improve the behavior of simulated vehicles in intersections [3]. However, they focus on the regulation possibilities offered by the norms, and not on their description capabilities. As for nondeterministic models, they are applied in various domains, from displacement models [10] to decision models in partially observable environments [5]. Nevertheless, no possibility to control the determinism of the mechanism is provided.

In this paper, we have presented a behavioral differentiation model for multi-agents simulations. It is composed of a data structure based on norms, and a generation engine based on nondeterminism. Violating behaviors can be created, and their deviations quantified. The engine allows controlling the determinism of the simulation at various levels, and is used to instantiate the behaviors. It provides various and consistent behaviors to the agents, two key elements for the realism of simulations. The variety is achieved by creating multiple and/or large definition domains for the norms, or with violations. The consistency is guarantied by the norms limits and enforced by the possibility to quantify the potential deviations. The approach has been applied to traffic simulation, using the driving simulation software SCANeR© II. The experimental results showed the interest of the model: we were able to increase the variety of drivers' behaviors, while guarantying the consistency of the parameters used.

References

[1] G. Boella and L. van der Torre. An architecture of a normative system: count-as conditionals, obligations and permissions. In *Intl Joint Conf AAMAS*, pages 229–231, 2006.

[2] E. Bou, M. López-Sánchez, and J. A. Rodríguez-Aguilar. Adaptation of autonomic electronic institutions through norms and institutional agents. In *Engineering Societies in the Agents World VII*, volume 4457 of *LNCS*, pages 300–319. Springer Verlag, 2007.

[3] A. Doniec, S. Espié, R. Mandiau, and S. Piechowiak. Non-normative behaviour in multi-agent system: Some experiments in traffic simulation. In *Intl Conf IAT*, pages 30–36, 2006.

[4] M. Esteva, J. Padget, and C. Sierra. Formalizing a language for institutions and norms. In *Intelligent Agents VIII*, volume 2333 of *LNAI*, pages 348–366. Springer-Verlag, 2001.

[5] L. Kaelbling, M. Littman, and A. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.

[6] B. Lacroix, P. Mathieu, and A. Kemeny. The use of norms' violations to model agents' behavioral variety. In *Coordination, Organizations, Institutions and Norms in Agent Systems, held with AAMAS'08*, pages 183–196, 2008.

[7] B. Lacroix, P. Mathieu, V. Rouelle, J. Chaplier, G. Gallée, and A. Kemeny. Towards traffic generation with individual driver behavior model based vehicles. In *Driving Simulation Conference North America*, pages 144–154, 2007.

[8] J. Maim, B. Yersin, and D. Thalmann. Unique instances for crowds. *Computer Graphics and Applications*, 2008.

[9] P. Noriega. *Agent mediated auctions: The Fishmarket Metaphor*. PhD thesis, Univ. Autonoma de Barcelona, 1997.

[10] C. Reynolds. Steering behaviors for autonomous characters. In *Game Developers Conference*, pages 763–782, 1999.

[11] J. Vázquez-Salceda, V. Dignum, and F. Dignum. Organizing multiagent systems. *Journal of Autonomous Agents and Multi-Agent Systems*, 11:307–360, 2005.