Journal of Simulation (2014) 8, 259–270

www.palgrave-journals.com/jos/

On the modelling of transportation evacuation: an agent-based discrete-event hybrid-space approach

B Zhang[1], WKV Chan[1]* and SV Ukkusuri[2]

[1]*Rensselaer Polytechnic Institute, Troy, NY, USA; and* [2]*Purdue University, West Lafayette, IN, USA*

This paper develops an agent-based discrete-event simulation (AB-DES) modelling framework for transportation evacuation by integrating an event scheduling scheme into an agent-based method. This framework has a unique hybrid simulation space that includes a continuous space and a flexible-structured network. This hybrid space overcomes the cellular space limitation in cell-based evacuation models and provides flexibilities in simulating different evacuation scenarios. Based on this AB-DES framework, we create an evacuation AB-DES model using the Parallel DEVS (Discrete-EVent System specification) formalism. We develop an algorithm to employ the event-scheduling approach to eliminate time-step scheduling used in classic agent-based models. Experimental results show that the AB-DES model is significantly more efficient than a pure ABS model while keeping high model fidelity and the same model capabilities including agent cognitive capability, collision avoidance, and low agent-to-agent communication cost. The agents' cognitive capability and autonomy property as well as the hybrid simulation space differentiate this AB-DES model from classic pure discrete-event models.
Journal of Simulation (2014) **8**(4), 259–270. doi:10.1057/jos.2014.3; published online 21 February 2014

Keywords: agent-based simulation; discrete-event simulation; hybrid simulation; DEVS; evacuation planning

1. Introduction

Transportation evacuation is an important issue concerning many researchers from different disciplines including engineering, social science, and policy. Evacuation planning is also a critical step in homeland defence preparation and disaster emergency management. Underestimation of this issue and ineffective planning could result in an over-simplified understanding of this complex evacuation issue, leading to loss of lives and properties. For instance, in 2005, when New Orleans was hit by Hurricane Katrina and the Houston area was threatened by Hurricane Rita, both evacuations suffered from tremendous traffic jams and chaos (Lindell *et al*, 2005). Similar traffic evacuations are common occurrences in many hurricanes. Understanding the traffic dynamics during an evacuation requires the modelling of interactions of a large number of agents and their interacting behaviour such as emergency agencies, first responders, household level behaviour, driver behaviour, and so forth. Various techniques have been proposed to model and understand the hurricane evacuation process (Hamacher and Tjandra, 2002; Lu *et al*, 2005; Zou *et al*, 2005; Chiu *et al*, 2008). Among these techniques, computer simulation has been an effective experimental means for evacuation planning and management due to its low cost and high speed. A real-time simulation model can help predict traffic conditions during an evacuation and identify bottlenecks in transportation networks. In addition, an evacuation simulation model allows the evaluation of various policies in improving the evacuation process such as contraflow lanes and lane closures. Therefore, a good simulation model can assist the development of a well-coordinated evacuation plan, which could save lives and properties.

An evacuation process is a stochastic dynamic process with complex interactions among autonomous evacuees, who employ various driving behaviours. The collective behaviour of the evacuees sometimes can be unpredictable or disastrous.

From a modelling point of view, to study such an evacuation process and to understand its intrinsic phenomena, one needs to model the behaviour of evacuees including their decision making, cognitive capabilities, and complex interactions among evacuees and between evacuees and emergency agencies. Classic discrete-event simulation (DES) models are not suitable for this type of systems as the entities in a DES model are rather simple, passive, and non-autonomous (see a discussion on this limitation of DES models in Chan (2010)).

ABS is a computational methodology for modelling intelligent agents, capturing system-level behaviours, and examining the interactions between autonomous agents. Because of the flexibility and capability in modelling agent behaviours and interactions, the ABS approach is particularly suitable for studying evacuation process to explore emergent collective phenomena during an evacuation. Several ABS models have been developed to investigate emergent evacuation situations (Hackney and Marchal, 2009; Lammel and Nagel, 2009; Zhang *et al*, 2009). A multi-agent behaviour-oriented model is developed in Kagaya *et al* (2005). There, a survey is first carried out to construct the behavioural rules of residents, and then a multi-agent evacuation

*Correspondence: WKV Chan, Rensselaer Polytechnic Institute, Department of Industrial and Systems Engineering, CII 5015, 110 8th St., Troy, NY 12180–3590, USA.

model is developed based on these rules. This multi-agent model is tested by reproducing a district earthquake evacuation in Japan. The work of Pelechano *et al* (2005) describes a crowd simulation system, which incorporates a psychological model to obtain emergent behaviours of humans during evacuation. This crowd simulation system manages individual motions and way-finding process and helps to understand human behaviours during building evacuations. A micro-simulation framework for large-scale pedestrian evacuations is introduced in Lammel and Nagel (2009). The simulation is an iterative process and every agent tries to optimize its individual evacuation plan at each iteration.

Although existing ABS models are able to capture the dynamics during an evacuation process and offer detailed analysis of agent interactions, they usually require intensive computational power. They usually employ a time-step based scheduling method to synchronize agent updates. In this time-step-based method, each agent updates its state at a fixed time step. This method is easy to implement but has a high computational cost. Selecting a suitable time step is therefore important (Hu *et al*, 2005). Large time step may result in incorrect simulation results (ie, missing events that should have occurred within two time steps), while a too small time step can result in unnecessary updates, and consequently, a high computational cost.

Although a pure DES model is not suitable for modelling complex interactions and behaviours of agents, some of the ideas in DES can be exploited to improve the performance of ABS models. In particular, a typical DES model uses a sequence of events to represent the changes of system state, which occurs at distinct time points. Different from the time-step-based method, a DES model executes based on events and only updates its system state when necessary (ie, occurrences of events). Nothing changes between events, and therefore any unnecessary state update can be avoided to save computation time. This event-based scheduling approach in DES is relatively efficient and will be exploited in this paper to create an agent-based discrete-event simulation (AB-DES) framework, which allows us to construct a rather efficient evacuation model compared with a pure-ABS model.

To facilitate the integration of DES and ABS, we employ a discrete-event modelling framework: Discrete-EVent System specification (DEVS). DEVS is a formal modelling and simulation framework derived from mathematical dynamical system theory and provides well-defined concepts for modular, hierarchical component construction, and reuse (Zeigler *et al*, 2000; Perez *et al*, 2010). It allows modellers to design and construct each model independently under certain protocols to facilitate the interaction between models. This event-based modelling approach has been widely used in many fields including transportation and emergency management (Ntaimo *et al*, 2004, Wainer, 2006, Sun and Hu, 2008). Most of the DEVS-based works in transportation use a cellular space, in which the space is discretized by a grid of cells. Although cellular space is a popular way of representing traffic flow models, it has certain limitations. First, it is not always easy to decide the size of cells. If the cell size is too small, a large number of cells will be kept in memory. Second, information diffusion and agent movements will be restricted to the Von Neuman (ie, four neighbours) or Moore (ie, eight neighbours) topology (Muller, 2009).

The integration of ABS and DES is a promising modelling methodology as it takes advantages of both simulation methods. Researchers have realized the benefits of ABS and DES and have used them together in various studies. Becker *et al* (2006) summarized the pros and cons of ABS and DES in modelling autonomous logistic processes. ABS can provide higher degree of flexibility and autonomy encapsulation while DES has a higher runtime performance. Warden *et al* (2010) developed a simulation middleware, PlaSMA, to analyse scenarios in logistics area. PlaSMA is a distributed DES, in which agents communicate with each other by message passing and simulation time advances in discrete steps of different length. PlaSMA adopts a conservative synchronization handled by the hierarchical simulation controllers.

A combined simulation system for rail/road transport is presented in Gambardella *et al* (2002). This system includes an agent-based planner for organizing inter-modal transport units and a discrete-event simulator to verify the feasibility of the plans and measure the performance. In Dubiel and Tsimhoni (2005), agent-based modelling is combined into a DES system with a test case of people moving in a theme park. The agent-based module handles people walking and interacting with other people and their environment, while the discrete-event module models the tram system in the theme park. Although both ABS and DES modules can exchange outputs with each other in these environments, the ABS and DES modules are constructed as two separate subsystems. Their event scheduling and execution are run separately. The integration of ABS and DES into a single model is not explored in these works.

Another line of effort expands existing DES framework to allow the flexibility and capability of agents (Zhou *et al*, 2006; Wu *et al*, 2008). For example, entities in DES are extended to mimic agents by adding more flexible attributes and behaviours. Communication protocols are also implemented to facilitate interactions among agents and between agents and the environment. However, the increasingly complex functionality and complicated interactions of agents can significantly degrade the performance of those extended DES models.

The work of Wagner (2004) refines the classical DES framework with the Agent-Object-Relationship (AOR) modelling language to provide a flexible object-oriented approach for agent-based modelling. However, in order to accommodate agent and environment updates, this simulation framework still applies a time-step-based scheduling method. A hybrid ABS for national airspace systems is developed and tested using different scheduling methods for synchronizing agent updates in Lee *et al* (2001). System accuracy and computational efficiency can be achieved by choosing an appropriate resynchronization interval. In general, a large resynchronization interval can be efficient but require good predictions on when resynchronizations are needed. Sometimes, a better prediction may require a high computational cost,

which could offset the saving gained by using a large resynchronization interval.

Building on existing hybrid modelling technique, this study develops an efficient and fully integrated agent-based discrete-event framework for evacuation and extend the cell-based environment of most existing evacuation models to a continuous-based environment, which allows flexibility in modelling and simulating movements of evacuees. In particular, we integrate the ABS approach and DES approach using a hybrid simulation space to capture the traffic behaviours and interactions between traveller agents. Different from existing DEVS models, our model uses a hybrid simulation space, which includes a flexible cell structure and a coordinate-based continuous space. This overcomes the limitations of restricted moving directions in cellular space DEVS models. The hybrid simulation space can better represent real traffic networks and reduce the number of cells in a model, thus increasing scalability.

We propose a queueing mechanism to handle agent interactions at intersections (see Section 3.2). We construct a global event list (which is a key component of the DES) to schedule all agent interactions (including collision avoidance) and other environment events. This global event list and associated event-handling mechanisms eliminate the time-step scheduling issue in classic ABS models. We note that although the time of the AB-DES approach advances based on events, this approach is different from classic DES models because it supports interactions of agents which are autonomous, pro-active, and cognitive, and the environment is hybrid discrete and continuous (see a discussion on the difference between ABS and DES in Chan (2010)). Nevertheless, because both ABS and DES are Turing complete, the AB-DES has the same theoretical modelling power as the DES (Chan *et al*, 2010). Therefore, the difference between AB-DES and DES lies in the high-level flexibility in modelling various agent behaviours, cognition, and decision. If the system is modelled by using a pure-DES approach, then it could require the use of a large number of artificial entities and complicated event scheduling. The resulting model is likely difficult to comprehend and inefficient. See an example in Chan (2010).

We design decision rules to simulate people's behaviour during evacuations. This AB-DES modelling approach can reduce a large number of updates by skipping unnecessary agent interactions, defined as those that cause no state changes in other agents or environment. Computational results show that the AB-DES model has a significant runtime improvement compared with the ABS model. A preliminary version of this modelling framework is presented in Zhang *et al* (2011). The current paper extends the model in Zhang *et al* (2011) by introducing prediction capability and conducting a more comprehensive experimental study to evaluate the performance of the model.

The rest of this paper is organized as follows. Section 2 describes the modelling approach and the detailed structure of each atomic model. In Section 3, we construct the atomic models in the Parallel DEVS formalism. Section 4 gives the performance evaluation of the model. Section 5 concludes the paper and presents future work.

2. Modelling approach

This section introduces the framework of the AB-DES approach. This framework allows most realistic traffic behaviours, including car-following, congestion, collision avoidance, intersection merging/crossing conflicts, etc. As a micro evacuation approach, this framework also allows various individual agent (traveller) behaviours to be included if needed. We use a queuing mechanism to handle agent interactions, thus keeping a low communication cost between agents in this framework.

2.1. Simulation architecture

Our integrated AB-DES approach focuses on the simulation architectures, and is not required to be written in a specific language. Figure 1 shows the modelling framework that integrates agents and a hybrid-space model for transportation simulation. This system is composed of traveller agents, a simulation space, and a simulation coordinator. The simulation space includes two components: a coordinate-based continuous space and a transportation network composed of roads and intersections. Each road or intersection is modelled as one entity. Roads and intersections are linked by directed ports as illustrated in Figure 1. Each traveller is an agent with its own decision rules and movement rules. During the simulation, a traveller can be anywhere in the hybrid-space, on a road, an intersection, or off-road locations. The simulation coordinator is the central control module of the simulation. It handles requests and sends commands to its subordinates. When a traveller is on a road or intersection, dynamic coupling between the traveller and the road or intersection is established and maintained by the simulation coordinator so that they can exchange information and interact with each other. Traveller agents can also travel in any off-road space that is not covered by the traffic network. The simulation coordinator manages such agents through the coordinate-based continuous space.

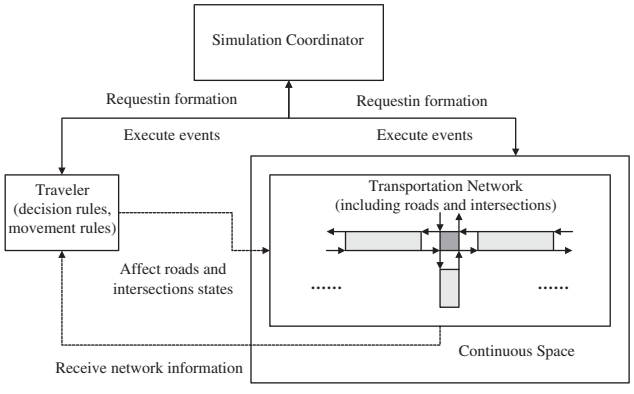

Figure 1 Model framework with agents and a hybrid space.

2.2. Transportation network modelling

The transportation network module is a coupled model composed of roads and intersections. Roads and intersections are atomic models linked by directed ports. Figure 2 (a) and (b) shows the schematic view and logical structure of a single-lane road, respectively. Each lane within the road is represented by one pair of in-port and out-port, which are connected to the corresponding intersections. Roads with multiple lanes can be built easily by adding more pairs of ports and lane changing behaviours need to be considered. In this paper, we apply a lane changing model which will be described in the next section. Each port can be seized by only one traveller at any time. The road model is coupled with the intersections during the initialization of the transportation network. When the simulation is running, the road model is dynamically coupled with the travellers by the Simulation Coordinator. In Figure 2 (b), the 'queryState' and 'outState' ports are built for communicating with other agents and the simulation coordinator. When queried, each road reports its state to the Simulation Coordinator.

Figure 3 (a) and (b) shows, respectively, the schematic view and logical structure of an intersection linking four single-lane roads. Similarly, additional pairs of ports can be added to represent multiple lane case. Also, there is no restriction on the number of roads to which an intersection can connect. The intersection model determines travellers' passing priorities based on their arrival times, in-ports, and out-ports. Similar to the road model, an intersection model can also report its state to the Simulation Coordinator.

2.3. Traveller agent modelling

Different from roads and intersections which are entities, each traveller is modelled as an autonomous agent with the abilities to decide its actions and to perceive the environment information.

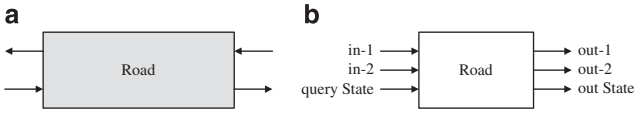

Figure 2 Structure of road atomic model for a single-lane road: (a) schematic view road; (b) logical structure (in-1 and out-1 represent the two ends of a lane entering and exiting a road, so are ports in-2 and out-2; queryState and outstate are communicating ports)

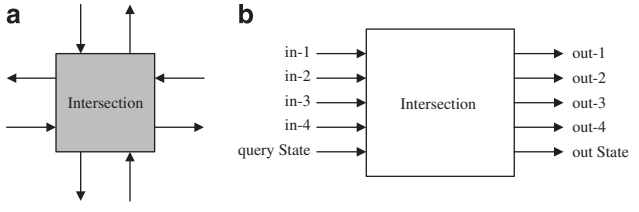

Figure 3 Structure of intersection atomic model.

A basic driver behaviour model for traveller agents is developed to handle how each traveller acts based on real-time travelling conditions. We implement a simple yet representative car-following model here. It calculates a vehicle's speed, acceleration rate, and deceleration rate considering the space from the vehicle in front. Based on the distance to the vehicle in front, the travelling process of a vehicle can be one of the three phases: free flowing, car-following, and decelerating, as suggested in Gazis *et al* (1959). For each traveller agent, the speed in the next time step is mainly determined by the legal speed of current road and the distance to the agent ahead. First, we define an upper bound and a lower bound for the distance. If the distance is larger than the upper bound, the traveller agent is at the free flowing phase. It does not interact with other vehicles and speeds up gradually to its desired speed. The acceleration rate is calculated based on the general car-following model introduced in Gazis *et al* (1959). If the distance is smaller than the lower bound, the traveller agent is at the decelerating phase. It has to slow down or stop to avoid collision. For deceleration, two functions are built, one for gradually decelerating and the other for emergent decelerating. When a traveller agent is approaching an intersection with a red traffic signal, it will also slow down until it completely stops. If the distance is in between, the traveller agent is at the car-following phase. It will adjust its speed based on the vehicle ahead. For roads with multiple lanes, different suitable lane changing models (Gipps, 1986; Moridpour *et al*, 2010; Bham, 2011) need to be incorporated based on testing map situation (urban, highway, etc).

Figure 4 shows the structure of the traveller agent model. Besides being queried and reporting its state to the Simulation Coordinator, a traveller can also receive travel times from roads or intersections to determine the occurrence time of its next action. When a traveller enters a new road, it sends update information to the Simulation Coordinator which will remove old couplings and add new couplings between the traveller and the new road.

Before an evacuation simulation starts, a traveller agent needs to decide its destination and evacuation route. It should be noted that different routing strategies can be incorporated in the simulation framework. In this paper, we focus on the simulation method itself.

After the evacuation route is determined, the traveller starts from its home and enters the transportation network. Because its home may not be located right at a main road, the traveller first needs to proceed to the nearest main road through a private path (such as a drive way) at a constant speed. Upon reaching a main road, the traveller can gradually speed up to its desired speed if the traffic condition allows. The travel time on each road can be computed using either a link performance function based on its

Figure 4 Structure of traveller agent model.

congestion level of the road (Sheffi, 1985) or random distributions using historical data.

Rules are built for travellers agents to handle the movement in the hybrid simulation space. Figure 5 shows an example of entering a road and moving on the road. We assume that the moving direction is from left side to right side. When a traveller agent enters the road from an intersection, three events will be used to represent the three phases (free flowing, car-following, and decelerating) during the travelling process. Also, if there is no other traveller ahead on the road, the traveller will continue on its free following phase until deceleration.

During the scheduling, the traveller needs to communicate with other travellers on the road. For example, a traveller in the car-following phase cannot reach the next phase earlier than its direct front neighbour, otherwise there will be a collision. Similarly, the scheduled leaving time will be delayed if the leaving port of the road is already seized at the scheduled time. If the road is too short for the traveller to fully accelerate, the travelling process will not be separated into three phases and the traveller will use a relatively low speed to pass the road.

Between leaving the previous road and entering the next road, a traveller must first seize the right of way before passing an intersection. When a traveller is coming to an intersection, the time to enter its next road must be determined. If the intersection is currently occupied by another traveller, a prediction function is used to estimate a passing time for the current traveller. If the intersection is still not available at the predicted time, then a prediction error happens and the passing time will be rescheduled. If multiple travellers with the same leaving port arrive from different ports with a very short time period, their arriving ports will be used to determine the right of way.

If a traveller starts from some private path or somewhere off road (continuous space beyond the traffic network) and needs to enter the road as shown in Figure 6, it will be directed to the road and request the Simulation Coordinator to add a dynamic coupling to the road. The traveller needs to communicate with other travellers on the road and get their estimated positions at the current time, and then the traveller can use this information to find an appropriate gap to enter the road.

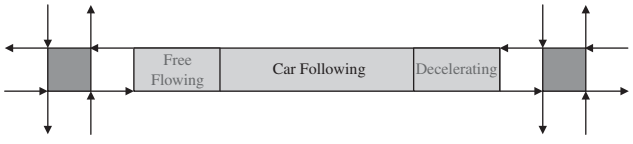

Figure 5 Moving within a road.

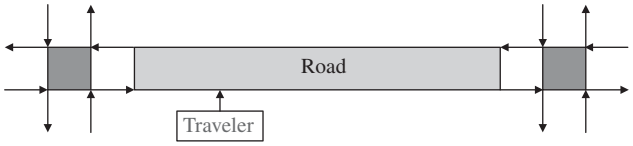

Figure 6 Entering a road from off-road.

Besides interacting with other agents while travelling on a road or passing an intersection, an agent can also receive environment information. For example, when a traveller agent is approaching a decision node (which could be an intersection), it may perceive the updated network information and adaptively choose its route. Depending on different information access level that agents can receive, multiple routing strategies can be modelled and tested. Furthermore, this AB-DES framework can also generate characteristics of agents and the environment. Statistic models can also be applied to simulate the departure timing and destination choice behaviours (see discussion in the future work). Here, in this paper, we focus on performance evaluation from a computational perspective.

3. DEVS expression

In this section, we express the model in terms of the Parallel DEVS formalism. A basic atomic model in the Parallel DEVS is defined in the following (Zeigler *et al*, 2000):

$$DEVS = (X_M, Y_M, S, \delta_{\text{ext}}, \delta_{\text{int}}, \delta_{\text{con}}, \lambda, ta)$$

where $X_M = \{(p, v) | p \in InPorts,\ v \in X_p\}$ is the set of input ports and values; $Y_M = \{(p, v) | p \in OutPorts,\ v \in Y_p\}$ is the set of output ports and values; S is the set of sequential states; δ_{ext}: $Q \times X_M^b \to S$ is the external state transition function; δ_{int}: $S \to S$ is the internal transition function; δ_{con}: $Q \times X_M^b \to S$ is the confluent state transition function; λ: $S \to Y^b$ is the output function; ta: $S \to R_{0,\infty}^+$ is the time advance function; $Q := \{(s, e) | s \in S, 0 \leqslant e \leqslant ta(s)\}$ is the set of total states.

To define the expression in DEVS, we introduce the following additional variables: *phase* is a control state used to keep track of the full states in the DEVS models; *tl* is the time instant at which the last event occurred; *tn* is the scheduled occurrence time for the next event with $tn = tl + ta(s)$; t is the global current simulation time; e is the elapsed time since the last event with $e = t - tl$; σ is the remaining time to the next event with $\sigma = tn - t$.

We adopt the format in Zeigler *et al* (2000) to present the DEVS atomic models in the following.

3.1. Road model

The road atomic model is defined using the Parallel DEVS as follows:

$X_M = \{(p, v) | p \in InPorts, v \in X_p\}$ with $InPorts = \{$'in-k', 'query State'$\}$ (k = 1, 2, …, n), $X_{\text{in-i}} = \{$TravelerID$\}$ (i = 1, 2, …), and $X_{\text{queryState}} = \{$queryValue$\}$;

$Y_M = \{(p, v) | p \in OutPorts,\ v \in Y_p\}$ with $OutPorts = \{$'out-k', 'outState'$\}$ (k = 1, 2, …, n), $Y_{\text{out-i}} = \{$TravelerID, leavingTime/NULL$\}$, and $Y_{\text{outState}} = \{$outValue$\}$;

$\delta_{ext}((phase, \sigma, inport), e, (p, v)) =$
  $(phase, \sigma, p, value(v))$ if $p =$ queryState
  $(phase, \sigma, p, leavingTime/NULL)$ if $p =$ in-i

$\delta_{int}(phase, \sigma) = (phase, \infty)$

$\lambda(phase, \sigma, inport, leavingTime/NULL) =$
  (outState, $value(v)$) if $inport =$ queryState

(out-i, *leavingTime/NULL*) if *inport* = in-i

$\delta_{con}(s, ta(s), x) = \delta_{int}(\delta_{ext}(s, 0, x))$

$ta(phase) = \infty$

When a traveller requests to enter a road, it can be either at an intersection or somewhere off road. If the traveller currently locates somewhere off road, we assume he/she is the only one who wants to enter the road from this location and no one is following him/her. If the traveller is at an intersection, a queue may emerge because of the incoming traffic flow behind the traveller. If the road has sufficient space to hold the traveller, it will enter the road instantly and receive an estimated leaving time computed based on the current road condition. If the road is too congested to hold the traveller, the traveller will receive a NULL message and wait in its current position. Since each intersection can only hold one traveller at one time and each port of the road can be possessed by only one traveller at any time, the current traveller who already holds the intersection is guaranteed to be the next one to enter the road when space becomes available. If the targeting road does not have enough space at current time, the traveller will be denied and hibernate. It will be notified when there is space available. If this traveller is currently on an intersection, this may trigger a delay in the estimated entering times of the travellers currently waiting in the intersection's waiting queue. In addition, further delays may occur on other roads connected to this intersection.

3.2. Intersection model

The intersection atomic model is expressed as follows:

$X_M = \{(p, v) | p \in InPorts, v \in X_p\}$ with $InPorts = \{$'in-k', 'query State'$\}$ (k = 1, 2, …, n), $X_{\text{in-i}} = \{$TravelerID, nextRoad$\}$, and $X_{\text{queryState}} = \{$queryValue$\}$;

$Y_M = \{(p, v) | p \in OutPorts, v \in Y_p\}$ with $OutPorts = \{$'out-k', 'out-State'$\}$ (k = 1, 2, …, n), $Y_{\text{out-i}} = \{$TravelerID, leavingTime$\}$, and $Y_{\text{outState}} = \{$outValue$\}$;

$\delta_{ext}((phase, \sigma, inport), e, (p, v)) =$

(*phase*, σ, *p*, *value*(*v*)) if *p* = queryState

(*phase*, σ, *p*, *outport*, *leavingTime*) if *p* = in-i and *phase* = 'vacant'

(*phase*, σ, *p*, *outport*, *predictedEnteringTime*) if *p* = in-i and *phase* = 'occupied'

$\delta_{int}(phase, \sigma) =$

('vacant', ∞) if *phase* = 'occupied'

$\lambda(phase, \sigma, inport, outport) =$

(outState, *value*(*v*)) if *inport* = queryState

(out-i, *leavingTime*) if *inport* = in-i and *outport* = out-i and *phase* = 'vacant'

(out-i, *predictedEnteringTime*) if *inport* = in-i and *outport* = out-i and *phase* = 'occupied'

$\delta_{con}(s, ta(s), x) = \delta_{int}(\delta_{ext}(s, 0, x))$

$ta(phase) = \infty$

The intersection model is different from the road atomic model because it allows multiple travellers coming from different roads to compete for entrance. We employ a queuing mechanism to queue the travellers as they arrive at an intersection and to determine the order for passing the intersection. We describe this queuing mechanism in the following.

The mechanism is similar to an all-way stop sign. First, the intersection atomic model maintains a waiting queue. When a traveller arrives at an intersection, the information of this traveller is stored into the queue according to the first come first served rule. In the case of simultaneous arrivals of two or more than two travellers, the right of way is determined using the travellers' travelling directions just like in the real network, for example, a traveller making a left turn must yield to a traveller going straight.

Second, each traveller waiting in the queue is given an estimated entry time to the intersection. Since there may be many travellers waiting to enter, it is inefficient to have all waiting travellers frequently check the availability of the intersection. A prediction function is applied here to estimate the entry time for a traveller based on the intersection's next available time, the order of the traveller in the waiting queue, and the time duration needed to pass an intersection. When a traveller requests to pass an intersection, if the intersection has space, it will enter the intersection and receive a leaving time, which is also set as the intersection's next available time. If the intersection has no (or insufficient) space, the traveller will enter the waiting queue and receive a predicted entry time to the intersection. Simulation results may be corrupted and need rollback if the predicted time could be too late. In order to avoid such issues, the prediction method is set to be conservative, which means there is a chance that the intersection is still not available at the predicted time (the predicted time is early). In such case, the traveller will need to call the prediction function another time. We call this a prediction error, which requires additional events to find the next available entry time. If the traveller is waiting on a road, this may trigger delay on other travellers following this traveller.

We note that this prediction function only predicts when a traveller can possibly cross an intersection. It does not reduce the number of necessary events. Therefore, the model fidelity is the same as that of a pure discrete-event model that considers only necessary events (ie, events that trigger other events).

3.3. Traveller agent model

Before expressing the traveller agent model in Parallel DEVS, we first introduce the following notations. Let 'FF', 'cf', and 'DC' denote, respectively, the free flowing phase, car-following phase, and decelerating phase. Notations 'onRoad', 'onInt', 'offRoad' indicate that a traveller is currently on a road, on an intersection, or off roads, respectively. Δt_{FF} and Δt_{DC} are the time durations that a traveller spends on the free flowing phase and decelerating phase, respectively. *c* is the time duration for a traveller to pass an intersection under a normal traffic condition. Other notations and variables are self-explainable by their names. The traveller agent model can be expressed in DEVS as follows:

$X_M = \{(p, v) | p \in InPorts, v \in X_p\}$ with $InPorts = \{$'travelTime', 'queryState'$\}$, $X_{\text{travelTime}} = \{$RoadID/IntID, leavingTime/NULL$\}$, and $X_{\text{queryState}} = \{$queryValue$\}$;

$Y_M = \{(p, v) | p \in OutPorts,\ v \in Y_p\}$ with $OutPorts = \{$'updateState', 'outState'$\}$, $Y_{updateState} = \{$add/remove/delay, RoadID/IntID$\}$, and $Y_{outState} = \{$outValue$\}$;

$\delta_{ext}((phase, \sigma, inport), e, (p, v)) =$

($phase$, σ, p, $value(v)$) if $p =$ queryState

('onRoad', c) if $p =$ travelTime and $v =$ (IntID, NULL)

('onRoad', $leavingTime$) if $p =$ travelTime and $v =$ (IntID, leavingTime)

('onInt', c) if $p =$ travelTime and $v =$ (RoadID, NULL)

('onRoad', $travelTime$) if $p =$ travelTime, $v =$ (RoadID, leavingTime) and

$travelTime < \Delta t_{AC} + \Delta t_{DC}$

('AC', Δt_{AC}) if $p =$ travelTime, $v =$ (RoadID, leavingTime) and

$travelTime \geqslant \Delta t_{AC} + \Delta t_{DC}$

$\delta_{int}((phase, \sigma, travelTime), e, (p, v)) =$

('CS', ($travelTime - \Delta t_{AC} - \Delta t_{DC}$)) if $phase =$ 'AC'

('DC', Δt_{DC}) if $phase =$ 'CS'

$\delta_{con}(s, ta(s), x) = \delta_{ext}(\delta_{int}(s, 0, x))$

$\lambda(phase, \sigma, inport, RoadID/IntID) =$

(outState, $value(v)$) if $inport =$ queryState

(updateState, $IntID$) if $phase =$ 'DC'

(updateState, $RoadID$) if $phase =$ 'onInt'

(updateState, ($addPort$, $RoadID$)) if $phase =$ 'offRoad'

$ta(phase, \sigma) = \sigma$

With AB-DES framework described above, we are able to use event-based scheduling instead of time-step-based method to execute the simulation in a hybrid simulation space, which includes a transportation network and a continuous space. Since agents are travelling on the transportation network, roads and intersections are the places where agent interactions occur. Agents communicate with each other through road and intersection models. This AB-DES model maintains necessary agent interactions to guarantee the correctness of the simulation logic. This model also has a low communication cost because interactions and events that do not cause state changes of other agents are avoided.

3.4. Simulation coordinator

The Simulation Coordinator model serves as the root coordinator in the Parallel DEVS formalism. It is the parent of all the travellers, roads, and intersections. The Simulation Coordinator maintains a global event list and handles the communication and interaction between travellers and the network. The global event list stores the scheduled events of the agents and the times at which the events are to occur. The scheduler algorithm in the Simulation Coordinator selects the next event from the global event list and executes it. If a traveller sends a message to request communication with others or if an event occurs and affects other travellers' states, the Simulation Coordinator will handle the information exchanges and schedule/reschedule corresponding events.

The Simulation Coordinator maintains all the necessary events and interactions among the agents and between the agents and the environment. This allows the model to execute without using the time-step scheduling method.

4. Performance evaluation

Most existing evacuation models are problem-specific and scenario-specific; they have different settings, assumptions, and scalability, in addition to being implemented using different platforms. All these diverse factors make performance comparison across different evacuation models almost impossible. In many cases, such a model-speed comparison attempt does not reveal the full potentials of different models that are designed for different scenarios. As such, the objective of this section is to illustrate the benefits of the AB-DES approach over the pure ABS approach.

We implement the AB-DES model under Java SE 7 Update 1. We compare the performance of this AB-DES model with a pure ABS evacuation model (Zhang *et al*, 2009), which is equivalent to the AB-DES model except that it applies the time-step scheduling method. The pure ABS model is implemented using Java under Repast framework, an agent-based modelling library (Repast, 2009). The AB-DES model and the pure ABS model are equivalent in the sense that they both model the same travel agent behaviours with identical inputs and they are implemented using the same programming language. Hence, the two models are comparable in both the simulation result and computational performance. The experiments are performed on a PC with an Intel Core 2 Due 2.4 GHz CPU and 2GB of memory.

We use a widely tested network, the Sioux Falls network, to evaluate the performance of our AB-DES model. The Sioux Falls network is good for examining data and debugging models (Bar-Gera, 2001). Figure 7 shows the structure of the Sioux Falls network. The square node is the destination. All circle nodes are the original evaluation source sites. The triangle nodes represent the intermediate transit sites. All the links have a single lane in each direction. Each experiment has the same number of traveller agents who enter the network from the origins and go through the nodes one by one. The Dijkstra's shortest path algorithm is applied to generate the route for traveller agents. The simulation ends when all the agents have reached the destination.

For all the experiments, the evacuation time is measured using a virtual simulation time unit. Therefore, the absolute value of this evacuation time is not of interest. Instead, we are interested in the difference of the virtual times between the AB-DES and ABS models, that is, the percentage of saving brought by the AB-DES model. To be comparable, the free flow travel time of each road segment are also set to be equal in both models. For example, if a traveller needs 15 min to go through a certain road in the pure ABS model, it also needs the same time to pass through the road in the AB-DES model. ABS and ABDES share the same free flow travel time.

In traffic simulation, it is crucial to select an appropriate number of replications to achieve a good balance between available resources *versus* acceptable results, especially for

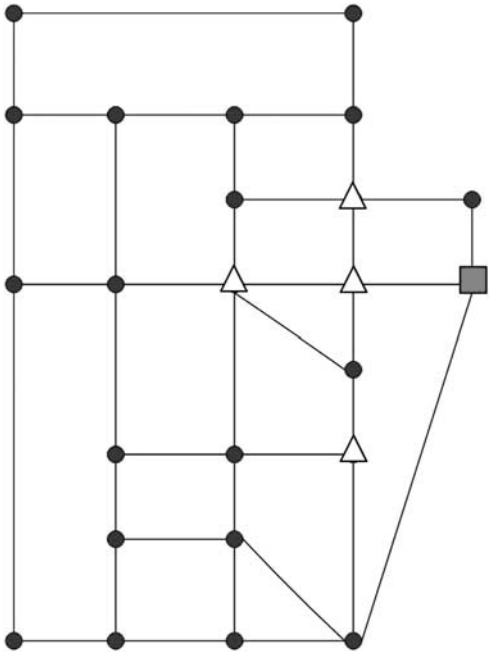

Figure 7 Sioux Falls network.

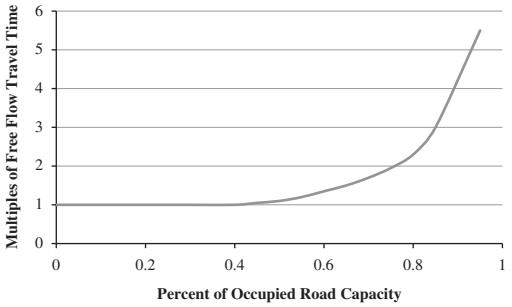

Figure 8 A typical link performance function.

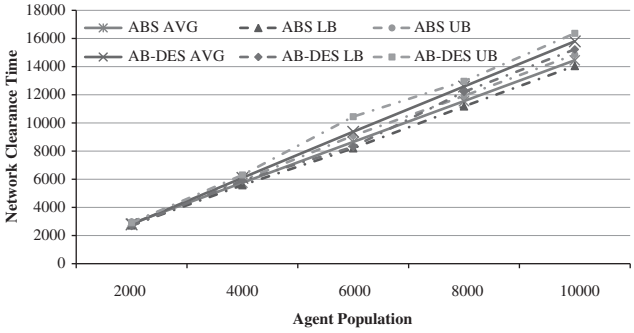

Figure 9 Network clearance time *versus* agent population.

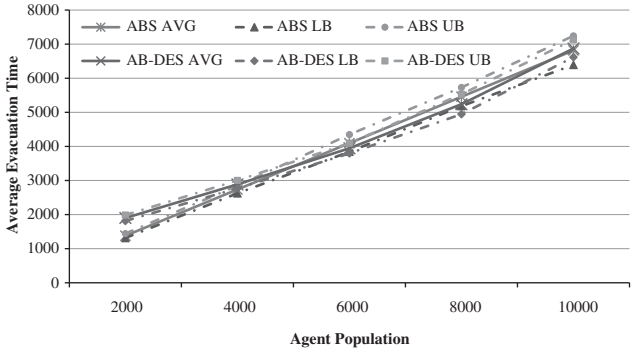

Figure 10 Average evacuation time *versus* agent population.

emergency simulation. There are a number of methods for selecting an appropriate number of replications to handle the randomness caused by different random numbers used (Diaz-Emparanza, 1996; Ahmed, 1999; Toledo *et al*, 2003; Mundform *et al*, 2011).

In this paper, to deal with the random variation of simulation results, we apply the method in Burghout (2013) to determine the number of required replications. We first set the initial number of replications to 20 and computed the average travel times of these 20 replications. A 0.05 level of significance is used and the estimated numbers of required replications for all simulated quantities are between 14 and 19 in both ABS and AB-DES. Therefore, we concluded that 20 replications is a reasonable number for our experiments.

In the AB-DES model, the travel time of each road is computed using a typical link performance function (Sheffi, 1985) shown in Figure 8. The free flow travel time is the time duration needed to pass through a road segment when there is no other vehicle on the road. When the traffic load on the road is relatively low, the travel time may still be close to the free flow travel time. The travel time increases slightly when about 40% of the road's capacity is taken. As the vehicle flow continues to increase, especially when the traffic load is close to the road capacity, the road travel time increases significantly.

We first compare the simulation results of both models. The network clearance time and the average evacuation time are used as indicators for result comparisons. The network clearance time is the time duration from the beginning of the simulation until all agents have evacuated. The average evacuation time is the average value of all agents' individual evacuation time.

Figures 9 and 10 show, the network clearance time and the average evacuation time under different agent populations, respectively. Roughly speaking, the two simulation models produce similar evacuation results, showing their equivalence. The slight difference between the evacuation times is due to using different random numbers and different ways of computing the travel times in the two models. In the ABS model, each traveller agent is scheduled to move toward the destination according to the evacuation route and traffic conditions at each time step. The travel time of each agent is therefore obtained through several time steps. In the AB-DES model, traveller agents use the link performance function to estimate the travel time on each road and advance to the next stage directly. Because the agents are not

uniformly distributed among the roads when travelling, some road segments may be quite congested. From the link performance function in Figure 8, we can see that the travel time increases rapidly when the road payload is close to its capacity. This could cause the estimated travel time longer than the actual travel time in the ABS model when the agent population is large. However, on average, the link performance function fits the actual travel time well. Therefore, the average evacuation times of both models are very close to each other as the agent population increases. Overall, given an identical simulation input, both models generate similar results. This also validate our AB-DES model.

Because the AB-DES model employs an event scheduling approach, each state update is an event by definition. To be comparable, in the ABS model, we consider an action taken by each agent at each time step as an event. Figure 11 shows the number of events and CPU time of the ABS model in changes of agent populations. As the number of agents increases, both the number of events and CPU time increase quadratically, while, in the AB-DES model, the number of events and CPU time only increase linearly as shown in Figure 12. The two models show a significant difference in their runtime performances. As to the CPU time, ABS model increases from about 900 s to 24 000 s, while the AB-DES model only increases linearly from 14 s to 70 s. Similarly, the number of events of the ABS model increases from about 3 million to 70 million while that of the AB-DES model only increases from 35 thousand to 174 thousand. The difference is expected to increase as more agents are simulated.

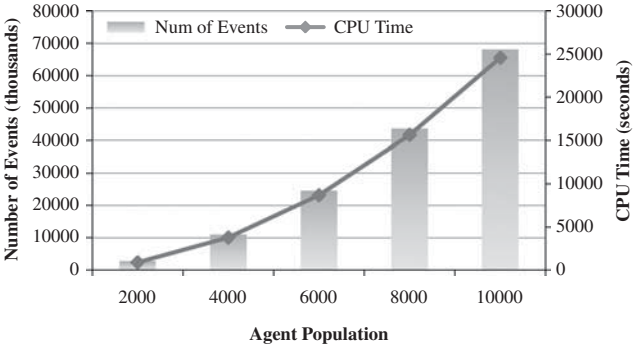

Figure 11 Number of events and CPU time of ABS.

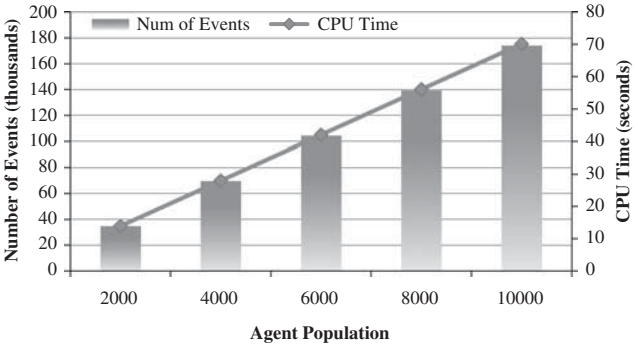

Figure 12 Number of events and CPU time of AB-DES.

It can be seen from Figure 13 that the total number of events in the ABS model is 80 times as many as that of the AB-DES model when the agent population is 2000. This number linearly increases to about 400 times when there are 10 000 agents in the network. The ratio of the CPU time of the ABS model to that of the AB-DES model reveals a similar pattern. When the agent number is 10 000, the AB-DES model is 350 times faster than the ABS model. At this trend of increasing, the runtime performance difference between the two models is expected to increase as more agents are simulated.

In the AB-DES model, the main computational cost during the occurrence of an event includes selecting the event with the smallest time label from the global event list and executing the selected event. If the total number of traveller agents is N, the cost of selecting an event is in the order of O(N). After an event is selected, the agent's next state and occurrence time are computed based on its current state using the state transition functions described in previous sections. If the agent is travelling on a road, it only needs to communicate with the agent directly ahead to avoid collision. Since we use a queuing mechanism to keep track of the agents in the transportation network, the communication can be done at a constant time. If the agent is going to enter a road from an intersection or enter an intersection from a road, it just needs to communicate with the road or intersection to check the space availability. In either case, the communication cost is constant. Therefore, the execution of the selected event usually takes constant time. However, some events may trigger a delay on other agents' event times and/or reschedule their events, which cause an additional O(N) time. Nevertheless, this situation does not happen frequently. Therefore, the average cost of executing an event in the AB-DES model is O(N).

In the ABS model, the event is the action taken by each agent at each time step. The execution process includes checking road condition, adjusting travel speed, and moving forward according to the route. When checking the road condition and adjusting speed, an agent needs to interact and communicate with other agents. The main computational cost lies in checking the status of other agents travelling on the same road as the current agent in order to adjust the current agent's travel speed and to avoid collisions. We assume there are m roads of single lanes for each direction in the network. The average number of traveller agents

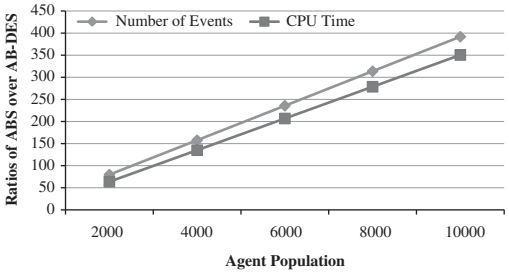

Figure 13 ABS/AB-DES ratios.

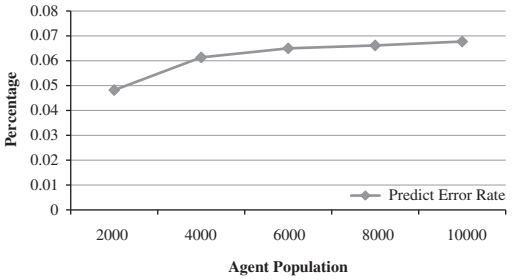

Figure 14 Prediction error percentage over event number.

on each road would be $N/2m$. Because the current agent needs to check the other agents on the road it is on, the time complexity is $O(N/2m) = O(N/m)$. In most cases, the total number of agents, N, is much larger than the number of roads m. Hence, the time complexity of the ABS model is approximately $O(N)$, with m neglected when N is much larger.

In general, the runtime of a simulation can be expressed by the product of the time per event and the total number of events. Although the time complexities per event of both models are at the same level, the runtime performance of the AB-DES model greatly outperforms that of the ABS model due to the significant difference in the total number of events. For example, when the agent population is 10 000, the number of events in the ABS model is about 400 times more than that of the AB-DES model because the AB-DES approach can reduce a large number of updates by skipping unnecessary agent interactions.

Figure 14 shows the percentage of prediction errors, that is, the number of prediction errors divided by the number of events during a whole simulation. As the agent population increases, the percentage only grows slightly from 5 to 6.8%. The performance of this rather simple and straightforward prediction method seems satisfactory. Although a more accurate prediction method could reduce this error, it is likely to require a higher computational cost, which would offset the benefits it brings.

Although animation is not a required component for a simulation model to execute correctly, it is a useful way of verifying and validating a simulation model. The AB-DES model trades some animation capability for a higher performance. In particular, it is not straightforward for the AB-DES model to generate smooth animation because the model proceeds by events (ie, it skips some of the updates that can make the animation smooth). Extra computations (ie, additional updates) are needed to generate smooth animation. In contrast, the ABS model executes actions and interactions of agents step by step and is able to provide a high-level detail of animation. The animation capability is one of the advantages of ABS method.

5. Conclusion

Despite its broad applications and capability in modelling complex systems, the conventional ABS approach usually suffers from the limitation of the time-step update mechanism, in particular when the environment is a continuous space. On the other hand, the DES method is relative efficient but could be inflexible in modelling and capturing autonomous behaviours of entities. This paper employs an event-based update mechanism to significantly speed up the corresponding ABS model while maintaining the same agents' cognitive capabilities and keeping a high model fidelity. Although hybrid ABS and DES models have been proposed in the literature, this model differs from existing ones in that it is a fully integrated AB-DES model that allows a hybrid continuous and discrete space environment and uses a network structure to keep track of agents and their interactions. This reduces the communication cost among agents. The coordinate-based continuous space gives the flexibility in representing real traffic situation and modelling various evacuation scenarios. The flexible cell structured transportation network allows the model to skip unnecessary updates that would otherwise be needed in conventional ABS models. As shown in the computational results, this approach can significantly reduce the CPU time, in particular, when the system is highly congested with a large number of agents in the network. Computational results show that the AB-DES model achieved a huge reduction in both the total number of events and CPU time as opposed to its ABS counterpart.

Like most traffic flow models, the speed of this model is a function of the model resolution, that is, a finer resolution would require a longer time to simulate. As mentioned in Section 4, this model has a resolution at the vehicle level. One can increase (or decrease) this resolution to speed up (or slow down) the model. Therefore, to circumvent this artefact, we compare the model speed relative to the pure ABS model instead of focusing on the absolute model speed. The relative saving in the event numbers is also used as a performance evaluation for the AB-DES approach.

While it would be interesting to compare the performance of the present model with other models should there exists one that also employs a hybrid continuous and discrete space environment (which could not be found), this paper makes no claim on the proposed model having the fastest execution speed. In fact, the model can run faster if the prediction method is improved, see future work below. The objective of this paper is to illustrate the integration of ABS and DES to achieve minimum event updates with high model fidelity and flexibility by allowing a continuous and discrete space environment.

As for future work, first, the prediction method in the AB-DES model could be improved. Although the current prediction method works well for the current experiments, the percentage of prediction errors still increases slightly as more traveller agents are simulated. Prediction errors can cause extra events and increase the total runtime. Although accurate prediction methods can reduce the total number of events, the extra computation for a better prediction method may not be justified. Therefore, one needs to balance the accuracy and cost of prediction when developing a better prediction method. Second, more traffic behaviours and evacuation strategies, such as lane-changing behaviour, can be incorporated to make the model more realistic. Finally, although the objective of this paper is to illustrate the

integration of ABS and DES to achieve an efficient evacuation model and to allow a hybrid space, validating this model is one important future work. For example, we can analyse hurricane data, population data, and calibrate the model using household decision making and timing data (Hasan *et al*, 2011, 2013; City-Data, 2012; Zhang, 2012).

References

Ahmed K (1999). *Modeling drivers' acceleration and land changing behavior*. PhD thesis, ITS Program, Massachusetts Institute of Technology, Cambridge, MA.

Bar-Gera H (2001). Transportation network test problems. Available at http://www.bgu.ac.il/~bargera/tntp/, accessed 18 June 2011.

Becker M *et al* (2006). Agent-based and discrete event simulation of autonomous logistic processes. In: Borutzky W, Orsoni A and Zobel R (eds) *Proceedings of 20th European Conference on Modelling and Simulation*, Bonn, Sankt Augustin, Germany, pp 566–571.

Bham G (2011). A simple lane change model for microscopic traffic flow simulation in weaving sections. *Transportation Letters: The International Journal of Transportation Research* **3**(4): 231–251.

Burghout W (2013). A note on the number of replication runs in stochastic traffic simulation models. Available at http://citeseerx.ist.psu.edu/viewdoc/download?rep=rep1&type=pdf&doi=10.1.1.216.7465, accessed 1 November 2013.

Chan WKV (2010). Foundations of simulation modeling. In: Cochran JJ (ed). *Encyclopedia of Operations Research and Management Science*. Wiley: New York, USA.

Chan WKV, Son Y-J and Macal CM (2010). Agent-based simulation tutorial—simulation of emergent behavior and differences between agent-based simulation and discrete-event simulation. In: Johansson B, Jain S, Montoya-Torres J and Yücesan E (eds) *Proceedings of the 2010 Winter Simulation Conference*, Baltimore, Maryland, USA, pp 135–150.

Chiu Y-C, Zheng H, Villalobos JA, Peacock W and Henk R (2008). Evaluating regional contra-flow and phased evacuation strategies for Texas using a large-scale dynamic traffic simulation and assignment approach. *Journal of Homeland Security and Emergency Management* **5**(1): 1–29.

City-Data (2012). Miami, Florida. Available at http://www.city-data.com/city/Miami-Florida.html, accessed 18 February 2012.

Diaz-Emparanza I (1996). *Selecting the number of replications in a simulation study*. Working Paper 1996-1. Available at SSRN: http://ssrn.com/abstract=1582 or http://dx.doi.org/10.2139/ssrn.1582, accessed 1 November 2013.

Dubiel B and Tsimhoni O (2005). Integrating agent based modeling into a discrete event simulation. In: Kuhl ME, Steiger NM, Armstrong FB and Joines JA (eds) *Proceedings of 2005 Winter Simulation Conference*, Orlando, Florida, USA, pp 1029–1037.

Gambardella LM, Rizzoli AE and Funk P (2002). Agent-based planning and simulation of combined rail/road transport. *Simulation Transactions of the Society for Modeling and Simulation International* **78**(5): 293–303.

Gazis DC, Herman R and Potts RB (1959). Car-following theory of steady-state traffic flow. *Operations Research* **7**(4): 499–505.

Gipps PG (1986). A model for the structure of lane-changing decisions. *Transportation Research Part B* **20**(5): 403–414.

Hackney J and Marchal F (2009). A model for coupling multi-agent social interactions and traffic simulation. *Proceedings of 88th Annual Meeting of the Transportation Research Board*, Washington DC, USA.

Hamacher HW and Tjandra SA (2002). *Mathematical Modelling of Evacuation Problems: A State of the Art*. Springer-Verlag Berlin: Berlin.

Hasan S, Mesa-Arango R and Ukkusuri SV (2013). A Random-parameter hazard-based model to understand household evacuation timing behavior. *Transportation Research Part C* **27**: 108–116.

Hasan S, Ukkusuri SV, Gladwin H and Murray-Tuite P (2011). A behavioral model to understand household level hurricane evacuation decision making. *ASCE Journal of Transportation Engineering* **137**(5): 341–349.

Hu XL, Muzy A and Ntaimo L (2005). A hybrid agent-cellular space modeling approach for fire spread and suppression simulation. In: Kuhl ME, Steiger NM, Armstrong FB and Joines JA (eds) *Proceedings of the 2005 Winter Simulation Conference*, Orlando, Florida, USA, pp 248–255.

Kagaya S, Uchida K, Hagiwara T and Negishi A (2005). An application of multi-agent simulation to traffic behavior for evacuation in earthquake disaster. *Journal of the Eastern Asia Society for Transportation Studies* **6**(303): 4224–4236.

Lammel G and Nagel K (2009). Multi agent based large-scale evacuation simulation. *Proceedings of 88th Annual Meeting of the Transportation Research Board*, Washington DC, USA.

Lee S, Pritchett A and Goldsman D (2001). Hybrid agent-based simulation for analyzing the national airspace system. In: Peters BA, Smith JS, Medeiros DJ and Rohrer MW (eds) *Proceedings of 2001 Winter Simulation Conference*, Arlington, VA, USA, pp 1029–1037.

Lindell MK, Lu J-C and Prater CS (2005). Household decision making and evacuation in response to hurricane Lili. *Natural Hazards Review* **6**(4): 171–179.

Lu Q, George B and Shekhar S (2005). Capacity constrained routing algorithms for evacuation planning: A summary of results. In: Medeiros CB, Egenhofer MJ and Bertino E (eds) *Proceedings of 2005 International Symposium on Advances in Spatial and Temporal Databases*, Angra dos Reis, Brazil, pp 291–307.

Moridpour S, Sarvi M and Rose G (2010). Lane changing models: A critical review. *Transportation Letters: The International Journal of Transportation Research* **2**(3): 157–173.

Muller JP (2009). Towards a formal semantics of event-based multi-agent simulations. *Multi-Agent-Based Simulation IX* **5269**(9): 110–126.

Mundform DJ, Schaffer J, Kim M-J, Shaw D and Thongteeraparp A (2011). Number of replications required in Monte Carlo simulation studies: A synthesis of four studies. *Journal of Modern Applied Statistical Methods* **10** (1), article 4. Available at http://digitalcommons.wayne.edu/jmasm/vol10/iss1/4, accessed 1 November 2013.

Ntaimo L, Zeigler BP, Vasconcelos MJ and Khargharia B (2004). Forest fire spread and suppression in DEVS. *Simulation-Transactions of the Society for Modeling and Simulation International* **80**(10): 479–500.

Pelechano N, O'Brien K, Silverman B and Badler N (2005). Crowd simulation incorporating agent psychological models, roles and communication. *Proceedings of 2005 International Workshop on Crowd Simulation*, Lausanne, Switzerland.

Perez E, Ntaimo L, Bailey C and McCormack P (2010). Modeling and simulation of nuclear medicine patient service management in DEVS. *Simulation-Transactions of the Society for Modeling and Simulation International* **86**(8–9): 481–501.

Repast (2009). Repast home page. Available at http://repast.sourceforge.net, accessed 5 March 2009.

Sheffi Y (1985). *Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods*. Prentice-Hall: Englewood Cliffs, NJ.

Sun Y and Hu XL (2008). Partial-modular DEVS for improving performance of cellular space wildfire spread simulation. In: Mason SJ, Hill RR, Mönch L, Rose O, Jefferson T and Fowler JW (eds) *Proceedings of 2008 Winter Simulation Conference*, Vols 1–5, Miami, Florida, USA, pp 1038–1046.

Toledo T *et al* (2003). Calibration and validation of microscopic traffic simulation tools: Stockholm case study. *Transportation Research Record* **1831**(1): 65–75.

Wagner G (2004). AOR modelling and simulation: Towards a general architecture for agent-based discrete event simulation. In: Giorgini P, Henderson-Sellers B and Winikoff M. (eds.) *Agent-Oriented Information Systems*, Springer-Verlag: Berlin, LNAI 3030, pp. 174–188.

Wainer G (2006). ATLAS: A language to specify traffic models using Cell-DEVS. *Simulation Modelling Practice and Theory* **14**(3): 313–337.

Warden T, Porzel R, Gehrke JD, Herzog O, Langer H and Malaka R (2010). *Towards Ontology-based Multiagent Simulations: The PlaSMA Approach*. In: Bargiela A, Azam Ali S, Crowley D, Kerckhoffs EJH (eds) *24th European Conference on Modelling and Simulation (ECMS 2010)*. European Council for Modelling and Simulation, Kuala Lumpur, Malaysia, pp 50–56.

Wu S, Shuman L, Bidanda B, Kelley M, Sochats K and Balaban C (2008). Agent-based discrete event simulation modeling for disaster responses. *Proceedings of 2008 Industrial Engineering Research Conference*, Vancouver, Canada.

Zeigler BP, Praehofer H and Kim TG (2000). *Theory of Modeling and Simulation,* 2nd edn. Academic Press: New York.

Zhang B (2012). *Agent-based discrete-event simulation and optimization of regional transportation evacuation*. PhD dissertation, Department of Industrial and Systems Engineering., Rensselaer Polytechnic Institute, Troy, NY.

Zhang B, Chan WKV and Ukkusuri S (2011). Agent-based discrete-event hybrid-space modeling approach for transportation evacuation simulation. In: Jain S, Creasey RR, Himmelspach J, White KP and Fu M (eds) *Proceedings of 2011 Winter Simulation Conference*, Phoenix, Arizona, USA, pp 199–209.

Zhang B, Ukkusuri S and Chan WKV (2009). Agent-based modeling for household level hurricane evacuation. In: Rossetti MD, Hill RR, Johansson B, Dunkin A and Ingalls RG (eds) *Proceedings of 2009 Winter Simulation Conference*, Austin, Texas, pp 2778–2784.

Zhou YH, de By R and Augustijn EW (2006). Explorative research on methods for discrete space/time simulation integrated with the event-based approach and agent concept. In: Wu H and Zhu Q (eds) *Proceedings of Geoinformatics 2006*: *Geospatial Information Technology*, Bellingham, pp D1–D11.

Zou N, Yeh S-T and Chang G-L (2005). A simulation-based emergency evacuation system for ocean City, Maryland during hurricanes. *Transportation Research Record* **1922**(1): 138–148.

Received 15 June 2013;
accepted 14 January 2014 after one revision