Agent-Based Simulation for Large-Scale Emergency Response: A Survey of Usage and Implementation

GLENN I. HAWE, GRAHAM COATES, DUNCAN T. WILSON and ROGER S. CROUCH
Durham University, England

When attempting to determine how to respond optimally to a large-scale emergency, the ability to predict the consequences of certain courses of action *in silico* is of great utility. Agent-based simulations (ABSs) have become the *de facto* tool for this purpose, however they may be used and implemented in a variety of ways. This paper reviews existing implementations of ABSs for large-scale emergency response, and presents a taxonomy classifying them by usage. Opportunities for improving ABS for large-scale emergency response are identified.

Categories and Subject Descriptors: I.6.1 [**Simulation and Modeling**]: Simulation Theory—*Model classification*; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Intelligent agents*; *Multiagent systems*

General Terms: Design

Additional Key Words and Phrases: Agent-based simulation, Emergency response

1. INTRODUCTION

1.1 Emergency Response and Preparedness

Multiple objectives may be used to define how to best respond to an emergency [Narzisi et al. 2006]. For example, in the U.K., the 'Concept of Operations' (ConOps) [U.K. Cabinet Office 2010b] lists thirteen general objectives for those involved in emergency response. High-level and qualitative, these include goals such as 'saving and protecting human life', 'relieving suffering' and 'protecting property'. Due to the potential for conflict, the 'Utopian' response, which simultaneously optimizes all objectives, does not necessarily exist. Consequently, Pareto-optimal trade-offs are the best that one can theoretically attain [Sawaragi et al. 1985]. The quandary of how to trade the objectives against one another [Branke et al. 2004] is complicated by the fact that "*their relative priority may shift as the emergency develops*" [U.K. Cabinet Office 2010b]. In the U.K., government ministers "*advise on the appropriate balance to strike in light of the circumstances*" [U.K. Cabinet Office 2010b].

Practically however, issues associated with emergencies such as time-constraints and uncertainty, mean that usually even these trade-offs are unachievable. The

Authors' address: School of Engineering and Computing Sciences, Durham University, England DH1 3LE.

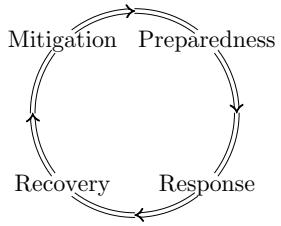

Fig. 1. The emergency management cycle.

response is the actions taken to: (1) avert an *imminent* emergency[1], mitigate its effects, prevent further damage, secure the scene, and address the immediate destructive effects of an incident (*crisis management*), and (2) prevent the impact from escalating (*consequence management*) [U.K. Cabinet Office 2010b]. These actions are precipitated by the bottom-up decision-making of the individuals involved, and the (often) top-down instructions which govern their coordination[2], neither of which are easily optimized: as Stirling [2003] says, *"no realistic decision making problem can account for all logically possible options"*. This is especially the case in *immediate impact* emergencies [London Resilience Team 2010] (such as transportation accidents and terrorist attacks), which occur with little or no warning, and thus have limited time for options to be considered.

At the operational level, even the premise of having well defined objectives may be a luxury during an emergency. Introducing the concept of 'flexecution', the flexible execution of plans whereby ill-defined goals are discovered and refined as they are pursued, Klein [2007a] explains that "*firefighters must handle emergent goals in the sense of managing goal conflicts but also revising goals depending on the progress they make*". Despite having changing, conflicting and emerging goals, an analysis of 69 incidents which the London Fire Brigade responded to found that *not once* was more than one alternative course of action considered by the incident commanders at any of the key decision points [Burke and Hendry 1997]. This is indicative of *naturalistic decision making* [Klein 2008], a description of how humans actually make *satisficing* [Simon 1957] decisions in demanding situations[3]. Whilst implying that 'optimal response' is an oxymoron, this satisficing (rather than optimizing) nature of human decision making "*should not change the standard that is sought. Indeed, failure to achieve the standard should serve as motivation to improve the ability to perform, rather than a rationale for lowering the standards*" [Stirling 2003].

[1]This is particularly relevant for '*rising tide*' [London Resilience Team 2010] emergencies with long '*lead-in*' times, such as flooding and health pandemics. Recent examples of responses to such emergencies include the evacuation of coastal areas in Hawaii in response to a tsunami threat [Reuters 2011], and the opening of floodgates in Louisiana to avert the potential flooding of New Orleans "*in a disaster that would have been much worse than Hurricane Katrina in 2005*" [USA Today 2011].

[2]In the U.K., the principle of *subsidiarity* states how these bottom-up and top-down forces should coexist: decision-making takes place at the lowest possible level, whilst coordination occurs at the highest necessary level [U.K. Cabinet Office 2010b].

[3]Klein et al. [1986] also observed this practice in fire-fighters.

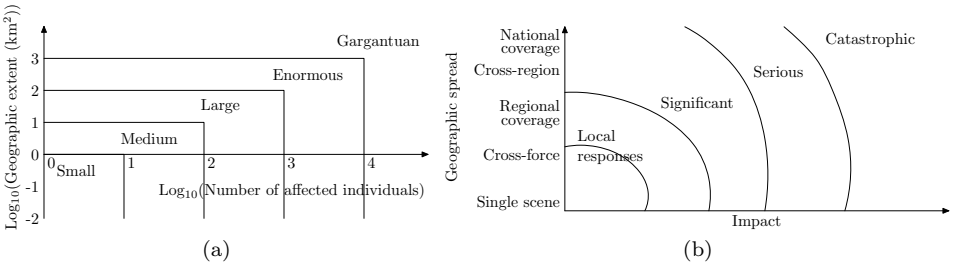

Fig. 2. Classification of emergencies using (a) Gad-el-Hak's Classification of Disaster Severity and (b) the U.K. Concept of Operations.

Activities which improve the ability to perform in response occur during preparedness, the phase of the emergency management cycle [Haddow et al. 2010] which immediately precedes response. Recovery, the fourth phase, follows response and feeds back into mitigation, the first phase, as shown in Figure 1. In the U.S., the Federal Emergency Management Agency (FEMA) National Preparedness Guidelines [FEMA 2007] describes preparedness as a cycle of *planning; organizing and staffing; equipping; training;* and *exercising, evaluating and improving.* Preparedness may even go beyond what is anticipated; for example, it may include tools for supporting *improvisation* [Mendonça and Fiedrich 2006; Mendonça 2007]. Work remains to be done however: for example, in his discussion of flexecution, Klein [2007b] notes "*We must develop new tools and support concepts to enable people to simultaneously define and pursue goals*".

Finally, we point out that some things may affect the efficacy of a response which are simply down to luck. For example, paragraph 153 of the coroner's inquest into the London bombings in July 2005 ('7/7') notes that "*... amongst the tragedy on 7/7, there were two fortuitous events. First, ... London's Air Ambulance was holding a clinical governance day. This enabled them to deploy a total of 27 physicians and paramedics across the four bomb scenes. Second, the explosion on the No. 30 bus occurred outside the British Medical Association building. This allowed many physicians to aid the victims of the attack.*" [Hallet 2011]. Whilst welcome, such good fortune should of course not be allowed to conceal any deficiencies in preparedness.

1.2 Large-Scale Emergencies

1.2.1 *Defining 'large-scale'.* Emergencies may be classified along many different dimensions, such as number of casualties, size of geographical area affected, duration, and cause. Different scales make use of different dimensions, according to their purpose. Examples include Berren's five dimensional classification [Berren et al. 1980], de Boer's disaster severity scale [de Boer 1990], Fischer's sociological disaster scale [Fischer III 2003] and Gad-el-Hak's classification of disaster severity [Gad-El-Hak 2009], which is illustrated in Figure 2(a).

Intuitively, when considering the dimensions relevant to classifying a generic emergency as 'large-scale', the 'large' may apply to either the size of geographical area affected, or to the 'impact' on the individuals therein, or both. Indeed,

these are the two dimensions used by Gad-El-Hak [2009], with 'impact' being the number of individuals displaced, injured or killed. They are also the dimensions used in the U.K. ConOps [U.K. Cabinet Office 2010b], where an emergency is classified as 'significant', 'serious' or 'catastrophic', depending on its geographical extent and 'impact' (two further categories are used for smaller emergencies which require only a local response). The 2005 7/7 London bombings (in which there were 56 fatalities and approximately 700 injuries), the 2007 U.K. floods (in which there were 13 fatalities and tens of thousands adversely affected), and the 2009 H1N1 flu pandemic (which lead to approximately 300 fatalities in the U.K., and hundreds of thousands infected) are all categorized as 'serious', leaving us to deduce that the generic 'impact' dimension is more likely to refer to the number of fatalities, than to the number of injured or adversely affected (which, across these three emergencies, varies by several orders of magnitude). The U.K. ConOps classification of emergencies is illustrated in Figure 2(b).

The two classifications in Figure 2 differ in at least two ways. First, in the ConOps classification, an emergency cannot be classified in either of the two most severe categories as a consequence of its geographical extent alone: a minimum level of impact is required as well. This is in contrast to Gad-el-Hak's classification, where geographic extent alone is sufficient. Secondly, as already discussed, in the ConOps classification, 'impact' seems to refer to the number of fatalities; Gad-el-Hak's 'impact' is measured by the number of individuals adversely affected. Nevertheless, the two classifications share some common fundamental features. In particular, both:

(1) classify an emergency by its geographical extent and a measure of its 'impact',
(2) have five categories of emergency, and
(3) both follow the general pattern that going to the top right of the scale increases the severity of the emergency.

Because of this concordance with practitioner usage (in the U.K. at least), we use Gad-el-Hak's classification to set lower limits on what is meant by 'large-scale', and thus set the scope of emergencies which will be considered in this paper. Thus, in this paper, a large-scale emergency is one which *either* affects an area of at least 10 km^2 in size, *or* affects at least 100 individuals, *or* both. Furthermore, we consider any emergency larger than this still as large-scale, i. e. we do not concern ourselves with whether or not an emergency is 'enormous' or 'gargantuan' (or 'serious' or 'catastrophic').

1.2.2 *Triggers of large-scale emergencies.* As well as the scale, the *cause*[4] of the emergency determines to a large extent the response required. As Dombrowsky [1995] states: *"For the state, the breakdown of public order and safety is the key, not the phenomena itemized. However, the specification of possible disasters is required because of the need for an appropriate selection of countermeasures to reestablish public order and safety. ... The trigger determines the measure; thus, riots the*

[4]The distinction between what is being responded to (the 'disaster' or 'emergency'), and what triggered it, is captured by Dombrowsky [1995]: *"Disasters do not cause effects. The effects are what we call a disaster."*

use of the National Guard; epidemics, the General Surgeon; terrorism, the Special Forces and Bomb squads. ... The type of phenomenon is only the key for the use of the appropriate tool box."

For example, in the U.S., FEMA [FEMA 2011] list seventeen types of emergency: *Chemical emergencies; Dam failure; Earthquake; Fire or wildfire; Flood; Hazardous material; Heat; Hurricane; Landslide; Nuclear power plant emergency; Terrorism; Thunderstorm; Tornado; Tsunami; Volcano; Wildfire;* and *Winter storms.* In the U.K., the Cabinet Office [U.K. Cabinet Office 2010a] list fourteen types of emergency, arranged into three different groups: **Natural events**: *Severe weather; Coastal flooding; Inland flooding; Pandemic human disease; Non-pandemic human disease; Animal disease.* **Major accidents**: *Major industrial accidents; Major transport accidents.* **Malicious attacks**: *Attacks on crowded place; Attacks on critical infrastructure; Attacks on transport systems; Non-conventional attacks; Cyber-attacks (data confidentiality); Cyber-attacks (infrastructure).* Each type of event has its own (location dependant) probability and impact associated with it. Together, these determine the overall risk it poses, and thus influence the amount of preparedness it attracts [FEMA 2007]. For example, in the U.K., the National Risk Register [U.K. Cabinet Office 2010a] plots the relative likelihood and impact of these fourteen types of emergency (at national level), and community risk registers refine these values at county level.

1.3 The Simulation of Large-Scale Emergency Response

1.3.1 *Methods of simulation.* The simulation of emergency scenarios is an integral part of preparedness [U.K. Cabinet Office 2011]. Simulations can range from pen and paper [Motowidlo et al. 1990] to real-life exercises. Whilst real-life exercises may offer a level of verisimilitude which is hard to surpass, this can come at a high cost. For example, the recent Orion training exercise [Orion 2010], which simulated the response to an earthquake measuring 8 on the Richter scale striking the U.K., cost €1 million, lasted 3 days, and involved hundreds of participants from across Europe [Crook 2010]. Such figures suggest that if some lessons can be learned via *in silico* simulations, especially for large-scale emergencies, then doing so makes economic sense. Furthermore, some things which are not even possible with real world exercises become possible via *in silico* simulation. Indeed, the ability to use "*simulations to re-produce scenarios that would be prohibitively expensive, dangerous, environmentally damaging, or even physically impossible to re-create in reality*" is listed as one of the main benefits of the 'Emergency Simulation Program', from the British Columbia Institute of Technology [Straylight 2010]. Another example is performing large parameter sweeps: by simulating the response to a suitably parameterized emergency scenario an appropriate number of times, each with different parameter values, we may build up a general picture of how different objectives depend on resource levels [Mysore et al. 2006], for example.

Computational simulation techniques which have been used for emergency response and preparedness include *systems dynamics* [Zhong and Kim 2011], *discrete-event simulation* [Connelly and Bair 2004], *stochastic modelling* [Mukherjee and Gupta 2009] and *queueing networks* [Au-Yeung et al. 2006]. For approximately a decade, *agent-based simulation* (ABS) has been used for emergency response and preparedness. It offers advantages over each of the methods mentioned [North and

Macal 2007]. Benefits of ABS include [Bonabeau 2002]: *ABS captures emergent phenomena; ABS provides a natural description of a system; ABS is flexible.* Emergency response satisfies the criteria for when it is suitable to use ABS [North and Macal 2007]. Many emergencies may involve the presence of crowds, e. g. [Samuelson et al. 2008; Oğuz et al. 2010]. In their discussion of tools for responding to large-scale emergencies, Lin and Manocha [2010] state that crowds "*are vital elements to model in a virtual environment*". In their review of simulation tools for crowd behaviours, Challenger et al. [2009a] conclude "*The most realistic simulation tools currently available on the market comprise agent-based models, and are populated by intelligent, autonomous agents, capable of making independent decisions and reacting to environmental conditions.*"

1.3.2 *Applications of computer simulation.* The appropriateness of computer simulation for emergency response and preparedness may be seen by observing its two major categories of usage [Ören and Longo 2008]:

(1) To provide experience for three types of training and entertainment:
   (a) by using virtual equipment (to enhance motor skills)
   (b) by gaming simulation (to enhance decision making skills)
   (c) by a mixture of real system and simulation (to enhance operational skills)
(2) To perform experiments, including for education, understanding and decision support.

For example, as part of their 'Integrated Emergency Response Framework', Jain and McLean [2003] identify the following five possible applications of computer simulation for the emergency response and preparedness domain:

Planning. Determining the impact of a disaster event, and the most appropriate ways in which to prepare and respond.

Vulnerability Analysis. Evaluating and assessing existing emergency response strategies to hypothetical emergencies.

Identification and Detection. Determining the possibility of a particular emergency occurring.

Training. Training emergency services personnel for handling emergencies through interactive simulations.

Real-time Response Support. Determining alternative response strategies, based on current knowledge of an unfolding emergency situation.

As the idealized use of computer simulation for emergency response and preparedness requires multiple phenomena to be modelled at the same time (for example, a fire simulation, a traffic simulation and a crowd simulation may all be required), Jain and McLean [2003] emphasize the need for a set of standards to allow different simulators to be integrated together.

Finally, from these five application areas, we note that users of computer simulation may include planners, trainers, and first responders. In principle, many organizations involved in emergency response and preparedness may benefit from the use of computer simulation, including non-governmental organizations providing humanitarian relief [Wolf 2003; Zhao et al. 2009].

1.4 Remainder of this Paper

In Section 2, we provide a description of agent-based simulation (ABS), beginning with terminology. After discussing the essential components of an ABS, and concepts pertinent to its application to large-scale emergency response, a brief overview of the two main toolkits, JADE and Repast, is given. Constructing an ABS for any application requires making design decisions. For example, the abstract concepts relating to agent architecture need to be made concrete. Also a realistic virtual environment needs to be constructed, such that the environmental state that each agent observes is sufficiently detailed for its purpose. This paper reviews how design decisions such as these have been made to date in the context of simulating large-scale emergency response. In Section 3.1, we begin by reviewing the different ways in which ABSs are used for large-scale emergency response, and propose a taxonomy classifying existing ABSs by usage. Validation and verification methods are also discussed. Sections 3.2 - 3.4 then review implementation details of ABSs for large-scale emergency response. Design decisions relating to the environment (Section 3.2), the agents (Section 3.3), and the large-scale nature of the problem (Section 3.4) are each reviewed. Section 4 concludes the paper with a summary, including future research directions regarding the use and implementation of ABS for large-scale emergency response.

2. AGENT-BASED SIMULATION

2.1 Terminology

We begin by acknowledging that what is now to be discussed (and is referred to as 'agent-based simulation' in this paper) suffers from ambiguous terminology [Hare and Deadman 2004]. In their review of 'agent based modeling toolkits', Nikolai and Madey [2009b] note the "*conflicting use of terms in different domains*", with 'agent'[5], 'agent-based' and 'multi-agent' having the most inconsistent usage. Even focussing on a single application area does not help a consensus be reached however, with *multi-agent system* [Gonzalez 2009a], *multi-agent simulation* [Massaguer et al. 2006], *multi-agent simulation system* [Takeuchi 2005] and *agent-based simulation* [Schoenharl et al. 2009] being just some of the terms employed in the context of large-scale emergency response, with no apparent pattern to their usage.

Attempts have been made to distinguish between such terms on different grounds. For example, Nikolai and Madey [2009b] used *heterogeneity* to make the distinction: an *agent-based system* is described as "*a system capable of modeling a large number of fairly homogenous agents*", whilst a *multi-agent system* is "*a smaller system of heterogeneous agents equipped with artificial intelligence*". Epstein and Axtell [1996] on the other hand describe the population of an *agent-based model* as 'heterogeneous', consisting as it does of "*distinct agents, each with its own genetically and culturally transmitted traits (attributes and rules of behavior)*". As well as heterogeneity, the emphasis in this definition is on 'distinct': there is no *aggregation* in agent-based models. Heterogeneity and lack of aggregation also form part of the emphasis in definitions of *individual based models*, a term which appears to be domi-

[5]The issue of what is meant by 'agent' is a perennial debate [Franklin and Graesser 1997; Drogoul et al. 2003; Petrie 2007; Castelfranchi 2010].

nant within the ecological modelling community. Due to its 'fuzzy' usage in the mid 1990s [Grimm and Railsback 2005], four biologically important criteria were proposed by Uchmánski and Grimm [1996] to distinguish individual based models from other 'individual-oriented' models, namely: (1) the degree to which the complexity of an individual's life cycle is reflected in the model; (2) explicit representation of the usage of resources by individuals; (3) discreteness (individuals are discrete, and thus population size is an integer); (4) heterogeneity among individuals. As well as heterogeneity and individuality, the emphasis placed on *interactions* has also been used to distinguish between terms. Sun [2006] describes a *multi-agent system* as "*a community of autonomous entities each of which perceives, decides, and acts on its own, in accordance with its own interest, but may also cooperate with others to achieve common goals and objectives*", and states that *agent-based social simulations* "*focus upon the interactions among agents*". Hare and Deadman [2004] also use the importance attached to modelling interactions (as opposed to modelling cognitive ability), and place various definitions in the literature on a "*continuum according to the types of interaction modelled*", with *agent-based modelling* appearing towards one end of the scale and *multi-agent simulation* appearing towards the other. The growing interest in including agents with cognitive ability in *agent-based social simulations* [Sun 2006; Gilbert 2006] however means that determining whether their focus is on modelling interactions or on modelling cognitive ability is difficult, as evidenced by two definitions of this term appearing in very different positions on the scale.

In this paper we shall follow Hare and Deadman [2004], and use 'agent-based simulation' (ABS) as an umbrella term, covering *most* papers which describe themselves using terms such as those above, but not all. As minimum requirements of an ABS, we refer to Epstein and Axtell [1996], who identify "*three basic ingredients: agents, an environment or space, and rules*". This is in concordance with Macal and North [2010], who state that "*a typical agent-based model has three elements*":

(1) A set of *agents*, their attributes and behaviours.
(2) A set of agent *relationships* and methods of *interaction*: an underlying topology of connectedness defines how and with whom agents interact.
(3) The agents' *environment*: agents interact with their environment in addition to other agents.

We stress at this point that the agents and environment are both *virtual*, so that by 'ABS' we mean software which runs on a desktop machine or supercomputer. This then precludes work such as [da Silva et al. 2008], which although uses a 'multi-agent system' for emergency response, does so through pervasive computing: agents are *real-world* decision making entities which exist in the *real environment*. Such work may be agent-based, but it is beyond the scope of this paper.

Furthermore, using the definition of 'large-scale' in Section 1.2, we restrict ourselves to ABSs which:

(4) Are *used to answer some question* pertaining to *large-scale* emergency response.

This further precludes some ABSs, such as DrillSim [Massaguer et al. 2006] (which investigates the effect of the number of floor wardens on the rate of evacuation of

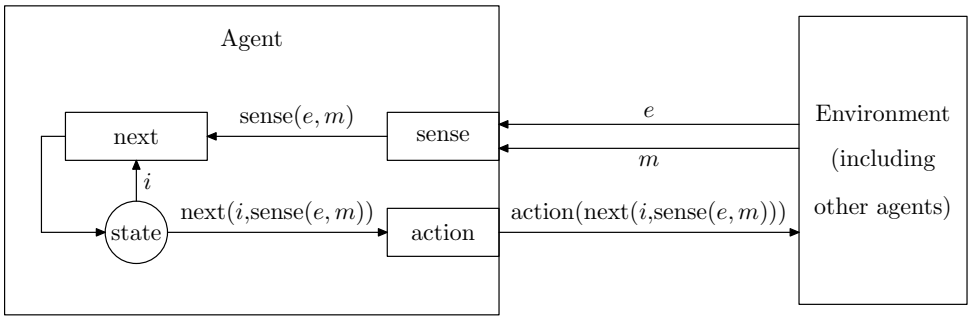

Fig. 3. An abstract agent with state, and its environment, adapted from [Wooldridge 2009].

28 agents from a single building), as the scenarios they simulate do not satisfy our notion of large-scale.

We now discuss briefly the central concepts of ABS, *italicized* in (1)-(4) above, in the context of simulating large-scale emergency response.

2.2 Agents

Perhaps the only trait common to every definition of an agent is autonomy [Wooldridge 2009]. An agent exercises its autonomy through actions in the environment in which it is situated. If the agent is intelligent [Wooldridge and Jennings 1995], then these actions are both proactive (initiated by the agent in order to achieve some goal) and reactive (in response to changes in the environment). Furthermore, communication between agents may also take place in order to achieve shared goals.

Figure 3, adapted from Wooldridge [2009], shows an abstract agent with an internal state, interacting with its environment, and with other agents situated within this environment. The internal state is updated via percepts, and used to determine which action to perform, via the following functions:

sense: $E \times M \rightarrow Per$. The agent observes its environmental state e and receives zero or more message(s) m from other agents, and generates a percept `sense`(e,m). E is the set of all possible environmental states, M is the set of all possible messages, and Per is the set of all possible percepts.

next: $I \times Per \rightarrow I$. The internal state i of the agent is updated via the `next` function, being set to `next`(i,`sense`(e,m)). I is the set of all possible internal states, and Per is as before.

action: $I \rightarrow Ac$. The `action` function programmatically represents agent behaviour. It selects and returns[6] an action, `action`(`next`(i,`sense`(e,m))), which is a member of Ac, the set of all possible actions available to the agent (which may include sending messages to other agents).

[6]Note that the return type of this function shares the same name as the function itself, *viz.* 'action'. To avoid equivocation, in this paper, when written in `typewriter` typeface, '`action`' refers to the function which algorithmically represents behaviour. This returns an 'action' (Roman typeface), an element of the set Ac which the agent performs. Specific actions, however, are themselves written in `typewriter` typeface. So, for example, we may refer to a `move` action, returned by an `action` function which programmatically represents the movement behaviour of an agent.

Whilst this description of an agent is quite common within the multi-agent systems community, it also covers the characteristics deemed *essential*[7] in the agent-based modelling community, namely [Macal and North 2010]:

(1) An agent is a *self-contained*, modular, and uniquely identifiable individual.
(2) An agent is *autonomous* and self-directed. It has *behaviours* that relate information sensed by the agent to its decisions and actions. An agent's information comes through interactions with other agents and with the environment.
(3) An agent has a *state* that varies over time. An agent's behaviours are conditioned on its state.
(4) An agent is *social* having dynamic interactions with other agents that influence its behaviour.

Given the application domain of large-scale emergency response, it is natural to consider an individual agent as representing one of the human individuals involved, typically either a civilian or rescuer. More generally however, agents are used to represent non-human entities as well: for example, vehicles and buildings have also been modelled as agents. Different implementations of agents for large-scale emergency response are reviewed in Section 3.3.

2.3 Environment

In the context of large-scale emergency response, the virtual environment in which the agents are situated represents a geographical area, which may be real or fictitious. Recall that the actions which an agent decides to perform depends partly on what it observes in its environment, i.e. what is contained in the environmental state e. Therefore all entities which are relevant to an agent's decision making, such as the buildings it observes and the road network it can travel along, should be represented in the environment, at an appropriate level of detail [Sato and Takahashi 2011]. Many of the types of emergencies listed in Section 1.2.2, especially the large natural events, involve some level of destruction to the environment. When this is the case, the environmental entities need to contain attributes which can be modified to reflect this.

One of the simplest methods of representing a geographical area is using a grid. When representing real areas, vector Geographic Information System (GIS) files are popular, but integrating them with ABS is not trivial [Gimblett 2002; Gonçalves et al. 2004; Schule et al. 2004; Rand et al. 2005; Brown et al. 2005; Castle and Crooks 2006; Pooyandeh et al. 2007; Sengupta and Sieber 2007; de Andrade et al. 2008; Crooks et al. 2008; Crooks 2008; Liebert et al. 2008; O´Sullivan 2008; Kennedy et al. 2009; Guo et al. 2008]. Representing the environment as a network may be sufficient, for example if there is no damage to the environment, as is the case with pandemics [Valle et al. 2006]. An example grid, GIS and network representation of a

[7]In [North and Macal 2007], Macal and North give slightly different criteria, listing *adaptiveness, the capability to learn and modify behaviour, autonomy* and *heterogeneity* as the "defining characteristics of agency". Furthermore, they state: "*In the case that a simulation is structured as an agent model, but the agents do not exhibit the requisite characteristics of agents as noted above, then it is not an agent-based model. Instead the agents in question are prototype agents, or 'proto-agents', and the simulation is a proto-agent model.*"

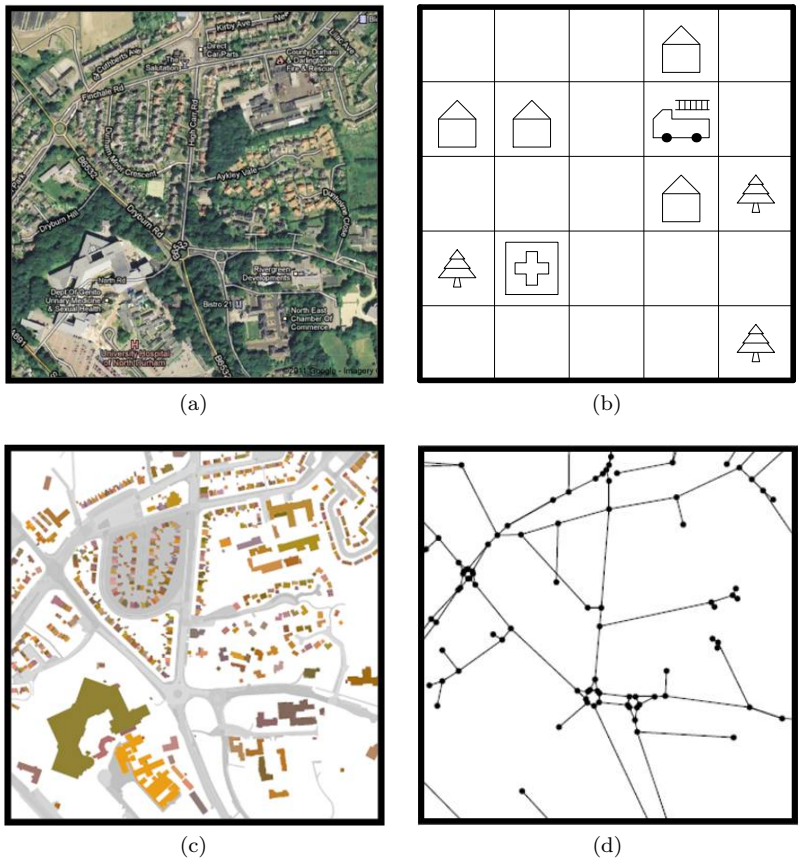

Fig. 4. (a) A geographical area and its representation using (b) a Grid, (c) vector GIS files and (d) a Network.

geographical area are shown in Figure 4. Different implementations of environments for large-scale emergency response are reviewed in Section 3.2.

2.4 Interactions

Epstein and Axtell [1996] describe "*three types of rules of behaviour for the agents and for sites of the environment*":

Agent-environment interactions. Each agent has a repertoire of actions which it can perform in its environment. The actions taken, which modify attributes of entities in the environment, and possibly of other agents, define the response. Actions are selected partly based upon what an agent senses in its environment, another form of agent-environment interaction. In order to allow the quick determination of the entities that an agent can observe, environmental entities need to be stored in a data structure which provides an efficient spatial index. Also, the environment may modify an agent's attributes: if an agent is in a building which collapses, for example, its health attribute will deteriorate.

Environment-environment interactions. The state of the entities in the environment may evolve due to processes other than agent actions. For example, fires may spread and buildings may collapse. These processes are the environment-environment interactions.

Agent-agent interactions. Agent-agent interactions include: communication (leaders may issue instructions, trapped agents may call for help); physical contact (adminstration of first aid, rescue from a building); social contact (being in the proximity of someone with an contagious disease may lead to infection).

These interactions form part of the discussion in Section 3.2 and Section 3.3.

2.5 Scale

When implementing agents and the environment, scale is an issue (from a computational perspective). As the application is *large-scale* emergency response, either the number of agents representing displaced, injured or killed human individuals is greater than one hundred, *or* the virtual environment in which the agents are situated represents a geographical area greater than ten square kilometres (in which case it potentially contains thousands of entities of interest), *or* both. Distributed memory parallelism is one way to achieve this scalability: a large geographical area may be divided into smaller sub-areas which are then simulated on separate processors, e.g. [Koto and Takeuchi 2003]. Even still, large numbers of agents may still exist in one sub-area (processor), and so shared memory parallelism may be required to enable them to act concurrently. Different techniques related to scale are reviewed in Section 3.4.

2.6 Usage

The ultimate purpose of each ABS reviewed in this paper is to help answer some question pertaining to large-scale emergency response. Each ABS does this by predicting, through simulation, the consequences of a particular course of action, with a particular level of resources, to a particular large-scale emergency, at a particular location. The question posed defines which aspects of the ABS may vary in order to optimize the response (as well as the objective function(s) being optimized), and which aspects are required to mimic reality.

From the real-world perspective, the response *is* the actions taken. These:

(1) are selected as a consequence of the behaviours of the real-world agents executing them; and

(2) are constrained by the resources available.

Thus, the actions which define the response may be altered by either changing agent behaviour, or changing the levels of resources, or both. In an ABS, determining optimal resource levels is a standard optimization problem, with resource levels as the design variables. When determining optimal agent `actions` however, there are two radically different approaches:

(1) *Optimize existing behaviours*: typically by either:
    (a) parameterizing current procedures in the `action` function of each agent,

and then determining the optimal parameter[8] values by applying a suitable optimization algorithm, e.g. [Narzisi et al. 2006]; or

(b) testing the consequences of alternative `actions`, when these alternatives exist in reality and the best choice is not clear, e.g. [Khalil et al. 2010].

(2) *Design and implement completely new procedures, which do not already occur in real life*: these can cover either bottom-up or top-down decision making, or both e.g. [Akin et al. 2010].

As put by Carley et al. [2006], ABSs containing agents of the second variety are[9] "*concerned with designing smart algorithms, not with investigating a current human social system as it exists and designing a public policy for it*" and thus are quite different from ABSs of the first variety, which are concerned with optimizing aspects of current agent behaviour (e.g. the setting of parameters which are subjective in nature, or choosing between alternative competing policies).

The earlier premise 'to a particular large-scale emergency, at a particular location' served to fix certain *control variables*, which characterize the severity of the emergency, and the environment that it is occurring in. If these control variables were to change, the optimal response may change too. For example, viruses which cause pandemics vary in how infectious they are. Thus the probability of a virus being transmitted from one individual to another, is a control variable when modelling a pandemic [Kim et al. 2008]. The optimal response, defined (say) by the optimal radius which quarantined areas should have, depends on this probability of transmission. Investigating how the optimal resource and behaviour parameter values depend on the values of the control variables helps determine the optimal response to new emergencies at new locations, through generalization. In some cases, the parameters relating to agent behaviour and resource levels are kept constant throughout, to mimic real-life. Then, by changing control variables, the effectiveness of current procedures and resource levels are tested in unprecedented scenarios.

Finally, the research question is often posed in order to learn lessons for potential future emergencies (i.e. as part of preparedness). However, ABS may also be used to answer questions about real emergencies, either retrospectively (learning lessons in hindsight, thus playing a role in preparedness for future similar events), or in real-time (as the emergency is happening, thus playing a direct role in the response). In all cases, some level of *verification* and *validation* needs to be demonstrated before an ABS can be trusted for use. Using definitions from the U.S. Department of Defense [Department of Defense 2009], verification is "*the process of determining that a model or simulation implementation and its associated data accurately represent the developers conceptual description and specifications*", whereas validation is "*the process of determining the degree to which a model or simulation and its associated data are an accurate representation of the real world from the perspective of the intended uses of the model*". As North and Macal [2007] state "*verification and validation work together by removing barriers and objections to model use. After*

[8]In practice, when behaviour is parameterized and optimized, resource levels are often parameterized and optimized too.

[9]The particular ABS being referred to here was RoboCup Rescue, which is discussed later in Section 3.

all, if a model runs perfectly well and reflects the workings of an important real-world system, why wouldn't someone want to use the model?". The serious nature of the application may even mean that some form of *accreditation* is requisite to usage. A review of different usages, including validation and verification, takes place in Section 3.1.

2.7 Agent-Based Toolkits

Many agent-based toolkits exist to ease the development of an ABS [Railsback et al. 2006; Nikolai and Madey 2007; 2009b; Allan 2010]. Bearing in mind the discussion on terminology in Section 2.1, the review by Allan [2010] is particularly interesting, as it distinguishes between toolkits intended for 'agent-based simulation and modelling' and those intended for building 'multi-agent systems'[10]. Given the number of toolkits available (Allan's review discusses 31 toolkits for agent-based simulation and modelling, and 13 for multi-agent systems, whilst the review of Nikolai and Madey [2009b] discusses 53 toolkits in total), even a cursory description of each is beyond the scope of this paper. In any case, in the domain of large-scale emergency response, only two toolkits appear to dominate[11]: JADE and Repast. Due to their popularity, and as they are referred to again later, we here give a brief description of each of these toolkits.

2.7.1 *JADE.* JADE (Java Agent DEvelopment Framework) is a popular toolkit for building multi-agent systems, which after a decade of development is now in its fourth major version [JADE 2010]. Using JADE, an agent is implemented by extending from the `Agent` base class, which itself implements the `Runnable` interface, and so is executed in a Java thread. Contained in this base class is some basic functionality common to all agents, such as the ability to send and receive messages (via the `send(ACLMessage msg)` and `receive()` methods respectively) and the scheduling of 'behaviours' (which cause agents to carry out tasks). Each agent behaviour is derived from one of the classes in the `Behaviour` class hierarchy. An agent may use multiple behaviours, and complex behaviours may be constructed from simpler ones. In addition, by using `FSMBehaviour`s, an agent may be modelled as a finite-state machine (FSM). An example JADE agent is given in Figure 5.

Further functionality may be added to JADE agents through the use of external Java-based software. For example, Jess [Jess 2008], a Java-based rule engine may be used to create intelligent agents, whilst Jadex [Pokahr et al. 2005] may be used to implement agents with the Belief-Desire-Intention (BDI) architecture [Bratman et al. 1988; Georgeff et al. 1999].

The code-base in JADE for the construction of agents make it a popular choice for emergency response ABS. However, as it is intended for developing multi-agent systems rather than agent-based simulations, other important aspects remain un-

[10] Allan's own words on the distinction between the two are: "*There is a fine line between agent based simulation and modelling (AMBS)* [sic] *and multi-agent systems (MAS). The former are used to simulate complex systems such as social networks and biology which exhibit emergent behaviours. The latter can also be used to simulate complex environments, such as supply chains. These could be referred to as 'smart' applications and components are called 'intelligent' agents*".

[11] It is interesting to note that, according to their respective home pages, JADE is for the development of 'multi-agent systems', whilst Repast is for 'agent-based modeling'.

```
1  import jade.core.Agent;
2  import jade.core.behaviours.*;
3
4  public class ExampleJADEAgent extends Agent
5  {
6    private int health_; // The health level of the agent
7
8    public void setup()
9    {
10       addBehaviour(new BasicBehaviour());
11   }
12
13   class BasicBehaviour extends SimpleBehaviour
14   {
15     public myBehaviour(Agent a)
16     {
17       super(a);
18     }
19
20     public void action()
21     {
22         ...
23         //e.g. modify health_ variable, set finished flag
24     }
25
26     private boolean finished = false;
27     public  boolean done() {  return finished;  }
28   }
29 }
```

Fig. 5. Example of an agent in JADE.

supported, in particular the construction of GIS-based environments. Toolkits for agent-based simulation, such as Repast, offer more functionality with regard to this aspect.

2.7.2 *Repast.* The Repast (Recursive Porous Agent Simulation Toolkit) Suite [Repast 2010] consists of two families of program for constructing agent-based models: *Repast Simphony* and *Repast HPC*.

2.7.2.1 *Repast Simphony.* Repast Simphony is a Java-based toolkit for building agent-based models [North et al. 2007]. It uses the Eclipse Integrated Development Environment (IDE), and models may be developed using either flowcharts (Repast Flowchart) [North et al. 2007] or a Java Application Programming Interface (API) (Repast Java).

Using the Java API, an agent does not have to extend any base class. Instead an agent is implemented as a Plain Old Java Object ('POJO'). Basic POJO agents are referred to as 'proto-agents' in the Repast Simphony documentation [Repast 2010]. Only when equipped with learning ability are they considered full 'agents'. Repast Simphony comes equipped with packages offering evolutionary algorithms and regression methods to enable the user to implement learning. Also available is a way to schedule one or more agent methods (representing behaviours) to execute.

Repast Simphony models time discretely, advancing it in units called 'ticks' [Crooks 2007]. Through the use of Java annotations, methods may be scheduled to execute, either at a certain frequency (every n ticks) or as a one-off (at tick m). Additionally, an agent action may be triggered as a result of another agent action.

Through the use of Geotools [Turton 2008] and the Java Topology Suite [JTS 2006], Repast Simphony offers methods for building an environment from GIS Esri (Environmental Systems Research Institute) Shapefiles [ESRI 1998]. This facilitates the construction of complex environments, containing potentially thousands of individual entities, such as buildings. Methods enabling agents to sense this environment are also provided. All agents in Repast Simphony belong to a 'Context', which is associated with one or more named 'Projections' (which may be Network, Grid, Continuous or GIS). The details behind these concepts are beyond the scope of this paper, but are explained in [Howe et al. 2006]. For an agent to `sense` what is in its proximity in a GIS environment, it can query its context for its GIS projection, use this projection to get its current coordinates, then call a `getObjectsWithin` method on the projection to determine what entities are within a certain rectangular region ('envelope'), which represents its field of vision.

Figure 6 shows skeleton code for an agent in Repast Simphony, belonging to a GIS projection named 'GIS'. It has a method called `step` which is scheduled to be called every time-step (starting at the first time-step). This calls a `sense` method, which queries the GIS projection for a list of the objects within a rectangular region around the agent. These can then be used to affect the state and decision making of the agent in methods called after `sense` (which would be inserted at line 9 in Figure 6).

As any Java class can be used as an agent (in the Java versions), JADE agents can be used in Repast [Yoo and Glardon 2009]. This is useful as it allows the strong agent functionality of JADE to be combined with the strong GIS-environmental modelling of Repast, however in practice (with the exception of Yoo and Glardon [2009]) one of the toolkits is always used exclusively. Finally, batches of simulations may be scheduled to run using Terracotta [Terracotta 2010], e.g. for parameter sweeps.

2.7.2.2 *Repast HPC.* Although Repast Simphony has been used to distribute single simulations across multiple machines [Gulyas et al. 2009], the difficulty in doing so means such usage has been uncommon. In December 2010, a separate toolkit specifically for the purpose of running single simulations on multiple machines, Repast HPC, was released [Collier and North 2011]. It is written using C++, and makes use of the Message Passing Interface (MPI) [Gropp et al. 2000] for distributed memory computing. Synchronization between processes is handled by a `RepastProcess` class, and is *conservative* [Fujimoto 2000]: that is, an event is not processed until all events scheduled to happen before it have been processed. The `RepastProcess` class is also the class used to request agents from other processes. Repast HPC uses the same concepts of 'context' and (distributed) 'projection' from Repast Simphony, however it does not yet support a distributed GIS projection. As the release of Repast HPC is so recent, it has not been used for emergency response simulations yet.

```
1  @AgentAnnot(displayName = ``ExampleRepastAgent'')
2  public class ExampleRepastAgent
3  {
4    // Schedule step() to start at tick 1, and repeat every tick
5    @ScheduledMethod(start = 1, interval = 1)
6    public void step()
7    {
8        Iterable<Object> percepts = sense();
9        ...//use percepts to update state and select action to perform
10   }
11
12   private void sense()
13   {
14     Geography<Object> geo = (Geography)context.getProjection(``GIS'');
15     Geometry geom = geo.getGeometry(this);
16     Coordinate coord = geom.getCoordinate();
17     double range = 10;
18     Envelope envelope(coord);
19     envelope.expandBy(range);
20     return geo.getObjectsWithin(envelope,Object.class);
21   }
22   ...//other methods
23 }
```

Fig. 6. Example of an agent in Repast Simphony, with a basic `sense` method.

3. ABS FOR LARGE-SCALE EMERGENCY RESPONSE

3.1 Usage

3.1.1 *Where, when, why and how ABS is used.* ABS is most commonly used during preparedness. Past real emergencies are often used as exemplar scenarios, in which case the usual procedure is first to tune the ABS to replicate how the real event unfolded, and then determine the optimal response in hindsight. Alternatively, the emergency being simulated may be without precedent. For a given location, preparation for an unprecedented emergency is warranted if its triggering event is plausible there. The plausibilities of triggering events varies at national and regional levels. For example, as mentioned in Section 1.2.2, the National Risk Register [U.K. Cabinet Office 2010a] describes fourteen high consequence risks facing the U.K., and plots their relative likelihood against their relative impact. Community Risk Registers then refine these relative likelihoods and impacts at county level. In the U.K. the greatest risks (in a Pareto-sense, in that they are non-dominated in terms of relative likelihood and impact) are 'attacks on transport', 'attacks on crowded places' and 'pandemic human disease', each of which ABS is well suited to modelling [Challenger et al. 2009b; Epstein 2009].

Unsurprisingly, the motivation for research within a country often depends on its own history of disasters, and the likelihoods of future potential disasters there. For example, six of the seven ABSs (we count WIPER and DADS together) in Figure 7 were used to simulate a large-scale emergency which has either occurred, local to where the ABS was developed, in recent history, or is a plausible risk, again local to where the ABS was developed, in the future. We use these seven ABSs to illustrate

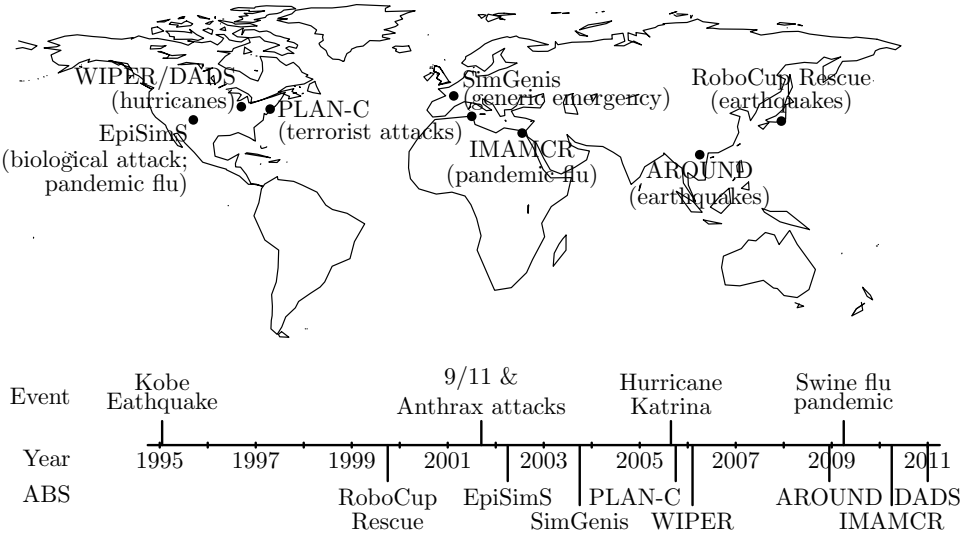

Fig. 7. Illustration of how the large-scale emergencies simulated using ABS varies around the world depending on the local risks and history of events.

the types of research questions asked, including which parameters are optimized (if any), what the objectives are, and what control parameters (if any) are varied to generalize results.

Along the Pacific Ring of Fire, research focuses on natural disasters. For example, various research programmes in Japan [Kitano et al. 1999; Takeuchi 2005] and Vietnam [Chu et al. 2009] have each focussed on the response to earthquakes. RoboCup Rescue [Kitano et al. 1999] is an ABS used to determine the optimal actions for police, fire brigade and ambulance agents, and their coordination centres, in the wake of a simulated earthquake. The 1995 Kobe earthquake was the original test scenario used [Takahashi et al. 2001], although more have been added over the years [Takahashi 2009]. The response is optimized through the design and implementation of better action selection methods. Research teams from around the world continue to test their action selection mechanisms on a variety of disaster scenarios[12], through an annual competition held since 2001[13]. The success of a response is measured using the following objective function:

$$S = \left(P + \frac{H}{H_{\text{init}}}\right)\sqrt{\frac{B}{B_{\text{max}}}} \tag{1}$$

where P is the number of alive individuals, H_{init} and H are the initial and final sums of health points for all individuals respectively, and B_{max} and B are the total area and final unburnt area for all buildings respectively. Clearly S is to be

[12]Using a variety of different scenarios represents a change in control variables.

[13]Further illustrating how research is driven by local risks, 12 out of the 13 entrants in the 2010 competition were from universities in earthquake-prone countries, namely China, Iran, Japan and Turkey.

maximized. Recognizing the multi-objective nature of emergency response, a score vector has recently been proposed [Siddhartha et al. 2009b; 2009a]. Using this, each individual objective retains its ipseity as a component of a vector, and the notion of dominance is used to compare different responses. As of 2010, the score vector has not yet been adopted for use in the official competition.

Also vulnerable to earthquakes is Vietnam. In 2007, a collaboration between French and Vietnamese universities resulted in an ABS called GAMA (GIS and Agent-based Modelling Architecture) [Amouroux et al. 2009] being developed. The following year the AROUND (Autonomous Robots for Observation of Urban Network after Disasters) project became one of its first applications [Chu et al. 2008], with an earthquake hitting Hanoi being the test scenario. The goal of the AROUND project is to capture mathematically agent behaviour which is consistent with the objectives of emergency responders. This is done by enabling rescue agents to learn their behaviour from the experts they represent, through interactive sessions. Thus, as with RoboCup Rescue, it is the behaviour which is optimized (rather than resource levels). However, there are two major differences:

(1) *Parameterizing the status quo versus the design of new procedures:* In AROUND, behaviour is modelled using a parameterized utility function, which is used for action selection; by varying the parameters, optimal behaviour is found[14]. In RoboCup Rescue, better responses are obtained by designing and implementing new action selection methods for rescuer agents, which don't necessarily bear any resemblance to real-life behaviour.

(2) *Difference in objectives:* In AROUND, optimal agent behaviour mimics that of the real-world rescuers they represent. In RoboCup Rescue, optimal agent behaviour maximizes the objective function in Equation 1.

The road map for RoboCup Rescue [Tadokoro et al. 2000; Tadokoro 2006] lists "*autonomous robot rescue agent team saves human lives*" as the goal for 2050[15]. If robots are tasked with actually carrying out the rescue, optimal behaviour is the target. The AROUND project also envisages the use of robots [Boucher et al. 2009], however for observation tasks only. The information gathered by these robots is fed back to a spatial decision support system (SDSS), which uses the ABS to predict the outcomes of possible courses of action (by human rescue teams). Boucher et al. [2009] state that "*ensuring the pertinence of the predicted outcomes requires incorporating in the system the knowledge used in situation by responders,*" hence the need for AROUND to capture existing behaviour: the ABS is used to predict how a particular response by human teams will proceed.

Whilst natural disasters are the main concern in the East, malicious terrorist attacks have been a motivating factor in the West. For example, following the terrorist attacks on the World Trade Center on September 11th 2001, and the

[14]More details on the implementation of agent behaviour in AROUND are given in Section 3.3.

[15]Before the arrival of autonomous rescue robots however, attempts are being made to transfer lessons from RoboCup Rescue to human responders. For example, Takahashi [2007] outlines a method whereby the behaviour of agents may be represented by a matrix. The components of the eigenvectors of this matrix give a basic interpretation of the agent behaviour, which may then be communicated to human rescuers.

anthrax attacks a matter of weeks later, the Center for Catastrophe Preparedness and Response (CCPR) was founded at New York University. As part of its LaSER (Large Scale Emergency Readiness) project, an ABS named PLAN-C (Planning with Large Agent-Networks against Catastrophes) was developed to predict what people, individually and collectively, will do in a large-scale emergency such as a terrorist attack, to aid preparedness for such situations [Mysore et al. 2006]. PLAN-C is an example of a parameterized model used to optimize existing resources and procedures.

In [Mysore et al. 2006; Narzisi et al. 2006; Narzisi et al. 2007], PLAN-C is used to simulate the first 50 hours following a Sarin attack on Manhattan island (unprecedented in the U.S.). Multiple experiments are performed, to investigate the effect of different parameters on the response. Some experiments only vary parameters related to resource levels. For example, the variation in the total number of fatalities is investigated in terms of:

—hospital resources levels (representing recoverable resources such as personnel and beds, and consumable resources such as drugs), and

—the number of first responders (Hazardous Materials teams (HazMats) and Major Emergency Response Vehicles (MERVs)).

Initial increases in either parameter leads to rapid decreases in the number of fatalities. However, beyond certain thresholds, further increases in either of these parameters had no discernable effect, with the number of fatalities fluctuating around a non-zero constant value. This demonstrated that a number of victims in the simulated Sarin attack were destined to die, regardless of resource levels.

Other experiments only vary parameters related to actions. Again, using the number of fatalities as the objective, optimal values of the following behavioural parameters are determined:

—when (i.e. the health level at which) a civilian should decide to go to hospital: if they go too early, hospitals can become overwhelmed; if they go too late, then the civilian risks death;

—when (i.e. the health level at which) a hospital should release a patient: patients may die if released too early; however, if released too late, then hospital resources are needlessly wasted;

—characteristics of civilian behaviour, in terms of 'worry' and 'obedience' parameters;

—the level of communication between hospitals (regarding their locations and capacities) and civilians: above a critical level of communication, the number of fatalities begins to increase as healthier victims learn of nearby available hospitals, which they then choose to travel to and reach before less healthy victims, who are then forced to travel further.

Responses are not generalized to account for any control variables. However the effect of the number of civilians present at the attack, the proportion of the population who are physically disabled, and the total number of hospitals in Manhattan, on the waiting times at hospitals is investigated. One counter-intuitive result was that removing the two nearest hospitals to the attack led to more victims being

treated earlier. This was attributed to victims distributing themselves more evenly over the remaining hospitals: before, large numbers of victims headed to the nearest hospitals, which quickly become overwhelmed.

Recognizing the multi-objective nature of emergency response, Narzisi et al. [2006] applied multi-objective evolutionary algorithms to the parameterized PLAN-C model to determine the Pareto-optimal trade-off between fatality rate, average health level of the affected individuals, and the average waiting time for victims at hospital. This time, only the first 16 hours following the attack were simulated. Ten parameters (two relating to the behaviour of civilians, five to the behaviour of hospitals, one to the behaviour of first responders, and two related to resources) were allowed to vary, although it is noted that it is difficult to actually control some of these in practice.

The Sarin attack modelled by PLAN-C is of a chemical nature, whilst the earthquakes modelled by RoboCup Rescue and AROUND represent physical destruction. Of quite different ilk are pandemics, which are of biological origin. EpiSimS, another ABS developed in the U.S. (at Los Alamos National Laboratory), has been used to investigate pandemics caused both maliciously and naturally: initially developed to simulate the spread of epidemics of natural cause in urban areas [Eubank 2002], it has since been applied to model the response to a smallpox attack in Portland, Oregon [Barrett et al. 2005]. EpiSimS is another example of a parameterized ABS being used to optimize existing resources and procedures. In the simulated smallpox attack, it was found that the number of deaths was most heavily influenced by how quickly infected individuals isolated themselves from the rest of society (whether by withdrawing to their homes through their own choice, or through actions by health officials). By comparison, the actual response deployed by officials to contain the spread of smallpox (mass evacuation of the city, vaccination, or quarantine) had a very small effect on the death rate.

Naturally caused pandemics are a worldwide threat. As mentioned in Section 1.2, the H1N1 flu pandemic (swine flu) which affected many different parts of the world in 2009 was classified as a 'serious' emergency in the U.K., and was certainly large-scale by any definition. Combatting a global pandemic requires effective response strategies across the world. The World Health Organization (WHO) recommends several response strategies to human pandemics, however different countries may take their own measures. For example, in Egypt, part of the response to the threat of swine flu involved the mass slaughter of all pigs in the country [BBC 2009] (against the advice of the WHO). Nevertheless, Egypt had the highest number of human fatalities in Africa as a result of the swine flu pandemic. Therefore it is unsurprising that the first application of an ABS developed at Cairo University, dubbed IMAMCR (Intelligent Multi-Agent Model for Crisis Response) in [Khalil et al. 2009], was to determine the optimal response to a flu pandemic in Egypt [Khalil et al. 2010]. IMAMCR is another example of optimizing existing resources and procedures. In this case, the procedures correspond to four real control strategies from the WHO. Success of each control strategy was measured in terms of the peak proportion of the population infected at any one time. Determining the optimal response involved simply identifying the control strategy which minimized this peak. 'Increasing awareness', which has the effect that individuals seek medical

help whenever they first experience flu-like symptoms, was found to be the most effective control strategy.

Of course, not every ABS takes motivation from a particular type of local risk. SimGenis [Saoud et al. 2004; Saoud et al. 2005; Saoud et al. 2006], for example, which was developed at Université la Manouba, in Tunisia, is designed to be 'generic per accident and location' [Saoud et al. 2006]. It was used in [Saoud et al. 2006] to design optimal, efficient and adequate rescue strategies, based on the initial state of victims, number of rescuers, and method of communication between rescuers. More precisely, the aim of the research was to determine how the response to a large-scale emergency depends on the use of:

—a centralized or a decentralized strategy, and
—an instantaneous electronic or a time-consuming paper-based means of communication.

The optimal response was generalized to different scenarios by varying the following control variables:

—the total number of victims,
—the initial states of victims, and
—the number of rescuers (doctors, nurses and fire-fighters)[16].

The main findings from varying these control variables were:

(1) "*there is no one unique 'best' rescue scenario* (strategy)*, . . . (as) this depends on the disaster configuration*", and
(2) "*for a given configuration, it is hard to predict which would be the best*", although some generalizations were able to be made[17].

Generalizing findings from particular scenarios to the generic case is important if results are to be applied quickly and reliably to unforeseen emergencies, once they occur. In SimGenis, this was done using a traditional 'design of experiments' style investigation. However, the use of machine learning methods to classify emergencies by characteristics of their optimal response, including possibly the use of transductive inference (reasoning from the particular to the particular, without constructing a generic case) [Vapnik 2006], may be another option for generalizing optimal responses.

The remaining ABS in Figure 7, WIPER, is designed for real-time response. WIPER (Wireless Integrated Phone-based Emergency Response) [Schoenharl et al. 2006; Schoenharl 2007; Pawling et al. 2008; Schoenharl et al. 2009] is a Dynamic-Data Driven Application System (DDDAS) [Darema 2004], which couples ABS with real-time data. It is composed of five main components [Pawling et al. 2008]. A *real-time data source* provides real-time data regarding cell-phone usage from cell-phone providers. Using a *historical data source* (a repository of normal cell-phone usage), a *detection and alert system* detects possible anomalies in cell-phone

[16]Strictly speaking, this is a parameter which can be varied, during preparedness. However, at the time a response is needed, this number is fixed, and so may be seen as a control variable.
[17]For example, when the ratio of victims to rescuers was high, a centralized strategy was found to be optimal; otherwise a distributed strategy was optimal.

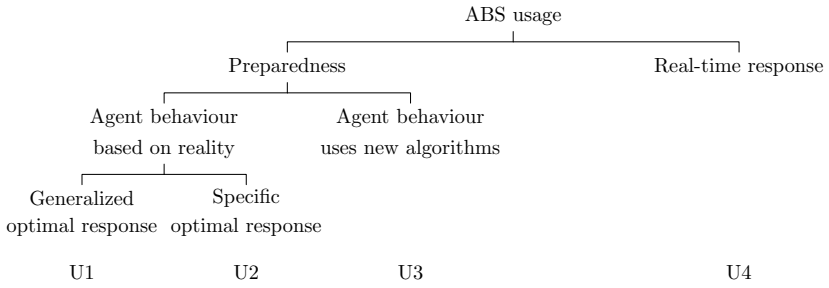

Fig. 8. Taxonomy of ABS usage for large-scale emergency response.

usage patterns. Hypotheses explaining these possible anomalies are produced by a *simulation and prediction system* (SPS), which are validated or rejected using streaming real-time data. Emergency response managers interact with these components through a web-based front end *decision support system*. The SPS operates by creating a batch of agent-based simulations, which is run in parallel. In each ABS, an agent's movement behaviour mimics that in one of three different types of crisis event: a *flee* event, where agents move away from a disturbance; a *flock* event, where agents move as a mob; or a *jam* event, where agents move towards their individual goals, but are constrained, as in a traffic jam. Using the real-time data source, the progression of each ABS is compared to that of the real-event. The ABSs with the greatest predictive power are identified, and used both to infer the cause of the anomalous behaviour, and make future predictions about civilian movements. A wider range of movement behaviours is supported in DADS (Dynamic Adaptive Disaster Simulation) [Chen et al. 2011], which builds upon WIPER using methods from continuum mechanics; Hurricane Katrina is cited as motivation for these developments.

From the seven ABSs discussed in this Section, we observe:

(1) ABSs may be designed for use:
   (a) before a large-scale emergency happens, during preparedness, or
   (b) during the real-time response (e.g. WIPER/DADS).
(2) Of those used during preparedness, an ABS may improve agent behaviour by:
   (a) varying some component of existing behaviour (e.g. PLAN-C), or
   (b) designing entirely new procedures, which are unrelated to existing behaviour (e.g. RoboCup Rescue).
(3) The description of the optimal response may be:
   (a) for a particular scenario (e.g. IMAMCR), or it may be
   (b) generalized, by investigating how this description depends on the control variables which describe the emergency and the environment (e.g. SimGenis).

Using these observations, we may categorize existing ABSs according to their usage using the taxonomy in Figure 8. The four taxa[18], labeled U1, U2, U3 and U4, are used to identify the usage of a variety of ABSs in Table I.

[18]In an attempt to have reasonable numbers existing in each taxon, creation of categories halts

Table I. The usage of various ABSs for large-scale emergency response.

Acronym (if any)	**Test scenario**	**Main references**	**Parameters** (if any) *Control variables, if any, are italicized*	**Objectives**	**Usage taxon**
no acronym	H1N1 flu outbreak in Kunming, China (2009)	[Wang et al. 2010]	Proportion of infected to quarantine; time between individual becoming infected and going to hospital	Peak number of infected humans (min)	U2
IMAMCR	H1N1 flu outbreak in Egypt (2009)	[Khalil et al. 2010]	Choice of control strategy: increasing awareness of population; vaccination; social distancing; quarantining	Peak proportion of the population infected at any one time (min)	U2
no acronym	Building evacuation	[Ren et al. 2009]	Number of security guards; velocity of evacuees	Evacuation time (min)	U2
no acronym	H5N1 flu outbreak in South Korea (2008)	[Kim et al. 2008]	Quarantine radius; *chicken incubation period; virus transmission probability*	Number of uninfected poultry farms (max)	U1
ResQ-Freiburg	Urban Search And Rescue in Bremen, Germany	[Kleiner et al. 2006]	not applicable	Expected number of survivors (max)	U4
PLAN-C	Food poisoning outbreak in Minas Gerais, Brazil (1998)	[Mysore et al. 2005]	Hospital resource levels; hospital-civilian communication level; number of hospitals; use of triage	Number of fatalities (min)	U2
	Sarin attack in Manhattan	[Narzisi et al. 2006]	Civilian: critical health level; unsafe health level; probability of having a communication device; phone update probability; On-site responder: dischargeable health level; Hospital: noncritical health level; dischargeable health level; low resource level; very low resource level; low beds level	Fatality rate (min); average health level of the affected individuals (max); average waiting time for victims at hospital (min)	U2
SimGenis	Generic emergency	[Saoud et al. 2006]	Communication: electronic or paper-based; Strategy: centralized or decentralized; *number of rescuers; number of victims; initial health state of victims*	Global evacuation time of all victims (min); rescue rate (max); death rate (min)	U1
BIOWAR	Anthrax attack at a sports stadium	[Carley et al. 2006]	*Mass of anthrax spores: 150g; 300g; 3000g*	Death rate (min); infection rate (min)	U2
EpiSimS	H5N1 flu outbreak in Los Angeles	[Valle et al. 2006]	Levels of anti-virals; use of vaccination or not; vaccination strategy (targeted/ uniform); effect of wearing masks or not	Number of new cases per day (min)	U2
	Smallpox attack in Portland, Oregon	[Barrett et al. 2005]	When infected people withdraw to home (early/ late/ never); delay in officials' response (4/7/10 days); vaccination strategy (targeted/ limited); mass evacuation of city; quarantine strategy	Number of deaths (min); number of infections (min)	U2
RoboCup Rescue	Earthquake in Kobe (1995)	[Takahashi et al. 2001; Skinner and Ramchurn 2010]	not applicable	Equation 1 (max)	U3

Verification techniques	
Structured code walk throughs	The developers of the ABS present the code they have written to an audience, which includes other developers (not involved in the development of the ABS), whose task it is to scrutinize the code.
Structured debugging walk throughs	This is similar to code walk throughs, but includes the debugging (execution of the code, stepping through it line by line), for a few test cases.
Unit testing	The automated testing of each function in the source code, as is done during extreme programming (XP) software development [Beck and Andres 2004].
Test cases and scenarios	The logging of information, particularly at the agent behaviour level, for a set of test cases, in a format which can then be scrutinized by the developers.
Visualization	Displaying the simulation output on a graphical display can help identify bugs, not readily identified using traces [Law and Kelton 1982].
Validation techniques	
Face validation	Asking domain experts whether the ABS behaves reasonably.
Retrodiction	Use historical data sets to test ABS predictions.
Prediction	Compare ABS predictions to outcomes from real events or field experiments.
Docking	Compare the results from an ABS to those from another model whose validity has been tested [Axtell et al. 1996].

Table II. Common techniques for verifying and validating an ABS.

3.1.2 *Verification and validation.* A large number of techniques exist for the verification and validation of computer simulations. For example, Balci [1997] provides a taxonomy of 77 verification and validation techniques for simulation models[19]. Some of the more common techniques for validation and verification of ABSs [Xiang et al. 2005; North and Macal 2007] are listed in Table II.

Saoud et al. [2006] discusses the use of three methods for the validation of SimGenis[20]. During *conceptual validation*, which aimed to validate the adequacy of the the underlying model in representing reality, experts from Service d'Aide Médicale Urgence (SAMU) were consulted to define model components. During *internal validation*[21], which aimed to ensure the correctness of the code, and *external validation*, which aimed to validate the adequacy and accuracy of SimGenis results with real world data, results from fieldwork (attending a 4 hour simulated plane crash, and discussing rescue plans with firefighters and medical staff) were used.

In PLAN-C, "*the model-checking approach focuses on individual agents' traces*" [Narzisi et al. 2006]. This is performed using external software, XSSYS Temporal Logic Trace Analysis System [Antoniotti et al. 2003]. Its use is demonstrated in [Mysore et al. 2005], where the time trace of a person clearly shows how their health state improves upon admittance to hospital. A time trace of a hospital also clearly shows how hospital resources decrease as the number of patients admitted

---

at an appropriate level, e.g.: It does not further categorize ABSs for real-time response, as there as comparatively few of them.

[19] A further 38 techniques are included in a second taxonomy for object-oriented simulation models.

[20] Three further methods, *cross-model, data* and *security* were mentioned, but not used.

[21] Internal validation is a synonym for verification [Bharath and Silverman 2010].

increases.

Feedback from domain experts (ambulance workers) is central to the development and use of AROUND. Consequently, much effort was put into the GUI design, a "*cognitive (and not simply cosmetic) necessity*" [Chu et al. 2009]. This may be seen as aiding face validation, which relies heavily on 'animation' and 'graphical representation' being presented to practitioners [Xiang et al. 2005].

Schoenharl [2007] describes three methods used to validate WIPER. Face validation was performed by the developers (rather than emergency services personnel). One criteria for WIPER was that different runs of the ABS with exactly the same input parameters (including seeds for random number generators) should give exactly the same results. This was confirmed in six validation tests, each of which ran four simulations with the same input parameters. Finally, simulated cell-phone activity was confirmed to correlate well with empirical data; results from statistical tests demonstrated "*significant evidence that the WIPER simulation generates call activity data similar to that seen in the empirical data*".

The validation of EpiSimS consisted of running (reduced size) experiments with twelve combinations of three input variables, namely the infectivity, the proportion of those who self isolate, and the number of people in the average room [Valle et al. 2006]. Variation of these variables all had the expected effect on the total number of people affected. IMAMCR was validated by comparison to a 'SIR' model (see Section 3.3 for description of 'SIR'); although not matching perfectly, the graphs of the proportions of the population who were 'susceptible', 'immune' and 'recovered' followed the same pattern for both models. Comparison to a 'SIR' model was also the approach taken in the validation of CROWD [Skvortsov et al. 2007] and BIOWAR [Chen et al. 2004]. Carley et al. [2004] give further detail of the validation of BIOWAR, including the use of a software tool called WIZER ('What-if AnalyZER') [Yahja and Carley 2006]. WIZER is a tool for automated validation, and consists of two main components: *Alert WIZER* and the *WIZER inference engine*. Alert WIZER detects if the outputs of the ABS is within acceptable bounds, by comparison to an empirical data source. Using this information, a database which relates model outputs to model parameters, and estimated bounds on the ABS model parameters (set by subject matter experts, in BioWar's case, experts on bioterrorism), the WIZER inference engine then identifies which ABS parameters to vary. These parameters are then tried in another ABS, and the process continues until some user-defined validation criteria is met.

If ABS is to be used by practitioners, the need for rigorous validation and verification is particularly important. For example, Takahashi [2007] reports feedback[22] received from local Japanese authorities on RoboCup Rescue, and states "*a greater validity of the ABSS is required to persuade fire-fighting departments to use them with other methods*". The use of ABS by practitioners will often require some form of *accreditation* [Duong 2010]. Accreditation is defined as "*the official certification that a simulation, or federation of simulations is acceptable for use for a specific purpose*" [Australian Government Department of Defence 2005], and may only follow appropriately high standards of verification and validation.

[22]Comments included "there are no theoretical backgrounds" and "the number of agents is small as compared to that in the real world".

Finally, we point out that some thoughts on the use of unvalidated large-scale agent-based models are presented in [Gross and Strand 2000].

3.2 Environment Implementations

3.2.1 *Representation.* When used for large-scale emergency response, the virtual environment in an ABS represents the geographical area affected by the emergency. In most cases this corresponds to a real-world area, but it may also be a fictitious area. For example, in the ABS SimGenis [Saoud et al. 2003; Saoud et al. 2004; Saoud et al. 2005; Saoud et al. 2006], the environment represents a fictitious city, of unknown size, 'including its routes and hospitals'. It is modelled as a grid of 'cells', where a cell is described as 'an elementary unit to represent graphically the environment' [Saoud et al. 2006]. Three types of cell exist: obstacle cells, danger cells, and normal cells. The type of a cell determines:

(1) *Which agents may pass through it*: no agents may pass through obstacle cells, and only fire-fighters may pass through danger cells.
(2) *The evolution of the health of any victim agents within it*: each victim's health state is modelled using a Markov chain; the probability of transition from one state to another depends, in part, on what type of cell the victim is in.

Modelling the environment as a grid restricts agent movement to up-down and left-right type movements. In SimGenis, Dijkstra's algorithm [Dijkstra 1959] is used by ambulance agents to determine the shortest route to a destination (presumably in this case each cell represents a node on a graph which resembles a regular grid).

A grid was also used to model the environment in an early version of PLAN-C [Mysore et al. 2005], which simulated a food poisoning outbreak in the Brazilian state of Minas Gerais in 1998. In both these examples, the authors recognize the inadequacies of using a grid to model the environment. For example:

—Saoud et al. [2006] concludes "*To be more realistic and in order to be able to simulate real accidents with real map, we are working on the connection of the simulator to a GIS*".
—Mysore et al. [2005] notes "*We might need to switch the environment to a real city. Transportation constraints and modes, roads, subways, and other geographical information might need to be incorporated*".

The use of vector GIS data files to construct a virtual environment has at least two main advantages over a grid (raster) based environment:

(1) The geometries of individual entities, such as buildings, may be described as polygons of arbitrary shape (as opposed to a collection of rectangular grids).
(2) Transport networks, such as road networks, which agents may travel along, may be represented accurately as a graph.

SimGenis was built using JADE [JADE 2010], which does not directly include capabilities for building environments from vector GIS data. This is perhaps why vector GIS environments in SimGenis did not materialize, and why the virtual environments in JADE-based ABSs are still often quite simplistic, e.g. [Gonzalez 2010]. PLAN-C however was built using Repast [Repast 2010], which does offer support for building environments from vector GIS data. In later versions of

PLAN-C [Mysore et al. 2006; Narzisi et al. 2006; Narzisi et al. 2006; Narzisi et al. 2007], vector GIS data were used to model Manhattan island, an area of size 60 km^2. After making some simplifying assumptions, open-source GIS files were used to construct a non-planar graph representing the road and (mostly underground) subway network. The 104,730 nodes of this graph included some 167 subway stops, and 30 hospitals [Narzisi et al. 2006; Narzisi et al. 2007] (22 hospitals in the earlier [Mysore et al. 2006]). In this case, a modified A* algorithm [Korf 1990] was used to determine agents' paths along the graph.

Other ABSs built using Repast also take advantage of its vector GIS capabilities. For example, GAMA [An 2008; Amouroux et al. 2009], the ABS which the AROUND project uses, is built using Repast. After preparation in ArcGIS [ArcGIS 2010], four shapefiles [ESRI 1998] (a vector GIS file format) representing roads, hospitals, police-barracks and agents, are used to model the Ba-Dinh district in Hanoi [Chu et al. 2008; Chu et al. 2009], an area approximately 10 km^2 in size. The road layer is used to construct a graph, along which the agents move. Shortest distances in this graph are computed in three different ways: 'on the fly', where each time an agent needs to know a shortest path to a destination node, it is computed using Dijsktra's algorithm [Dijkstra 1959]; 'on the fly with memory', same as 'on the fly', but if the shortest path between two nodes has already been computed, it is not recomputed; and 'pre-computation', where all shortest paths are computed before the beginning of the simulation using the Floyd-Marshall algorithm [Floyd 1962; Warshall 1962].

In order for an ABS to construct rapidly the widest range of environments possible, the use of GIS data which are open-source and commonly available is desirable [Takahashi et al. 2005]. Movements towards this goal have been seen, for example in RoboCup Rescue, where early environments were not built using standard GIS files[23]. Thus, initially, new environments appeared quite slowly:

—In 2001, the original environment included with RoboCup Rescue represented the Nagata Ward in Kobe city, Japan, centred on the JR Nagata Railway. Representing an area of 0.25 km^2, this contained 778 buildings, and a road network with 818 edges and 765 nodes [Takahashi et al. 2001].

—In 2002, a 'Virtual City' map was added. This represented a fictitious area of 0.29 km^2, and its road network contained 622 edges and 531 nodes.

—In 2003, a map of Foligno, Italy was added. Representing an area of 0.74 km^2, this contained 1078 buildings, and a road network with 1480 edges and 1369 nodes.

—In 2004, a map generator was introduced, which enabled the creation of random maps.

It was not until 2009 that the rapid construction of environments representing real cities was enabled in RoboCup Rescue. Then, as part of the 'infrastructure' competition[24], Gobelbecker and Dornhege [2009] introduced a plug-in for the JOSM

[23]Chu et al. [2009] describes RoboCup Rescue's *'lack of support for standard GIS description'* as a *'major problem'* and cites it as a reason for the construction of GAMA.

[24]The RoboCup Rescue 'infrastructure' competition is a separate competition to the one mentioned on page 18. Its purpose is to drive development of the RoboCup Rescue ABS itself, rather

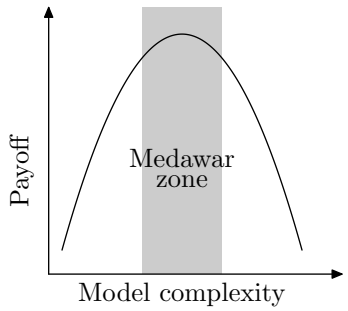

Fig. 9. Payoff of bottom-up models vs their complexity; here we view complexity as complexity of an ABS environment.

(Java OpenStreetMap) editor [JOSM 2010], which enabled the creation of files for use in RoboCup Rescue from OpenStreetMap [OpenStreetMap 2010]. Where OpenStreetMap did not provide building data for an area, this tool automatically created buildings, based on the space available. Sato and Takahashi [2011] compare environments built using this tool with environments built using a combination of open-source data from OpenStreetMap and the Japanese Geographical Survey Institute (GSI), which provides accurate geometries for individual buildings. Using the more accurate GSI data, the number of individual buildings constructed increased by a factor of 6-8.5, whilst the average floor area of each building decreased by a factor of 7-10. In some scenarios this was found to influence simulation results: the larger buildings produced using the tool of Gobelbecker and Dornhege [2009] took a longer time to burn than the smaller buildings produced using the more accurate GSI data; furthermore, gaps between smaller buildings prevented fire from spreading.

It should be noted however, that increasing the level of detail, and thus complexity, of the environment is not a desideratum *per se*. Rather, an optimal level of complexity will exist at which the environment should be described. In other contexts, this has been described as the 'Medawar zone' [Grimm et al. 2005], and is illustrated in Figure 9. If the environment contains all the information possible, the model becomes overcomplex for its purpose; if insufficient information is included, to the point that simulations are inaccurate, then the model becomes oversimplified. The 'Medawar zone' represents that intermediate level of complexity where payoff is maximum. Further research, such as investigating the effect of further dividing buildings into rooms, as is done in [Lee et al. 2008], would be useful in identifying the optimal level of complexity needed for modelling environments for large-scale emergency response.

3.2.2 *Environment-environment interactions.* The spreading of fire (leading to the burning of buildings) mentioned earlier is an example of a process which makes changes to the environment, independent of the actions of agents. These processes model environment-environment interactions. The presence of such processes means

than agent action selection mechanisms.

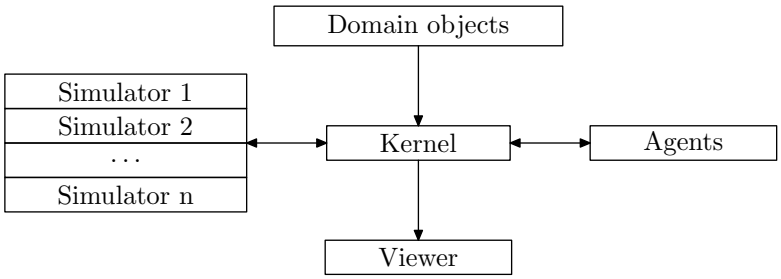

Fig. 10. Architecture of the 2010 RoboCup Rescue Simulation Platform [Skinner and Ramchurn 2010]. 'Domain objects' uses GIS files to construct the environmental entities. The 'simulators' are responsible for modelling the dynamic parts of the environment.

that the state of the environment may change whilst an agent decides upon a suitable action. For an agent in such a scenario, the environment is said to be *dynamic* [Russell and Norvig 2009]. Conversely, if the state of the environment does not change whilst an agent deliberates, it is said to be *static*. In a dynamic environment, the extra processes present may be simulated:

(1) by means of additional programs, which communicate with the ABS; or

(2) as part of the ABS itself.

The use of additional programs to evolve the environment leads to modularity: additional programs may be added in the future, and existing programs may be removed if required, with minimum impact on the remaining code of the ABS. In RoboCup Rescue for example, these extra programs are known as *simulators*, and they are responsible for modelling fires [Nussle et al. 2005], crowds [Brenner et al. 2005], traffic and most recently floods [Shahbazi et al. 2010]. These simulators communicate with the entities represented in the environment, which are contained in a separate *domain objects* component, via TCP/IP sockets, through a central *kernel*. Further *agents* and *viewer* components also communicate through this kernel. The overall architecture of RoboCup Rescue [Skinner and Ramchurn 2010] may be seen in Figure 10.

Alternatively, the processes controlling the dynamism of the environment may be an integral part of the ABS code itself. This is often the case with ABSs built using Repast. For example, in PLAN-C [Narzisi et al. 2006] , *'it is possible to initialize multiple catastrophe-agent and setting their time of activation'* [sic]. Thus, special types of agent in PLAN-C are used to make the environment dynamic. From the point of view of the rescuer agents, the environment is dynamic, as its state can change independently of their actions.

Not all view an ABS as the most appropriate way to model the environment and its evolution during an emergency. For example, Gonzalez [2009b] states that although a "MAS" may be used for modelling responders, it is "*not necessarily appropriate for modeling the crisis situation itself. Other types of models may be required for representing the objects, events and dynamics of the environment in*

which those agents interact."[25]. One of the reasons given for this is that entities in the environment "*differ from the response agents in that they do not necessarily require a high degree of autonomy or complexity.*" As a solution, the use of a *discrete-event simulation* package [Jacobs et al. 2002] is proposed for modelling the environment and its evolution, whilst the ABS models the agents. Practical software issues arose when combining the JADE built multi-agent system with a discrete-event simulation. For example, 'animated proxies' of the agents needed to be represented inside the discrete-event simulation, to model their interaction with the environment. Furthermore, to allow communication between agents and the environmental entities, an object containing all the environmental entities needed to be created which both inherited from a JADE class and implemented an interface from the discrete-event simulation package.

Examples of environment-environment interactions include:

Fire. Nussle et al. [2005] proposes a simulation of fire spread based on direct heat transport, radiation, and convection, which has been implemented in RoboCup Rescue. Ruas et al. [2009] propose the use of the Swarm toolkit [Swarm 2011] with this fire simulation.

Flood. Shahbazi et al. [2010] describe a flood simulation for use in RoboCup Rescue, whilst Tanigawa et al. [2005] describe the use of a flood model in the distributed simulation IDSS [Koto and Takeuchi 2003].

Weather. Carley et al. [2004] describe the use of wind in BioWar; this is important as it can disperse biomaterial at the time of a biological attack.

3.2.3 *Providing information to `sense`.* Finally, from a software point of view, ABSs which use GIS environments require a suitable data structure for storing entities. In particular, an efficient index is needed, to determine the entities situated in an agent's vicinity. RoboCup Rescue uses an R-tree data structure [Guttman 1984; Manolopoulos et al. 2003], from the Java Spatial Index [JSI 2010] package, to store entities in the environment, and provide fast access to them. Repast makes use of the GeoTools package [Turton 2008], which provides R-tree and quadtree [Finkel and Bentley 1974] data structures for spatial indexing. The use of such indices occurs in the `sense` method of agents; the remaining agent concepts are discussed in the following Section.

3.3 Agent Implementations

In Section 2.2 we saw that, at an abstract level, an individual agent is composed of a *state*. After `sense`ing its environment, an agent's state is updated by a method called `next`. An `action` method then uses the updated state, to select an action for the agent to perform in the environment. In this section, we shall review how these abstract concepts have been made concrete in ABS for large-scale emergency response.

3.3.1 *State and `next`.* The state of an agent may consist of multiple parameters and data-structures. We have already seen one agent parameter explicitly in

[25]Part of the reason Gonzalez raises this issue, whereas e.g. the PLAN-C authors do not, is that Gonzalex uses JADE (a 'multi-agent' toolkit) whereas PLAN-C is built using Repast.

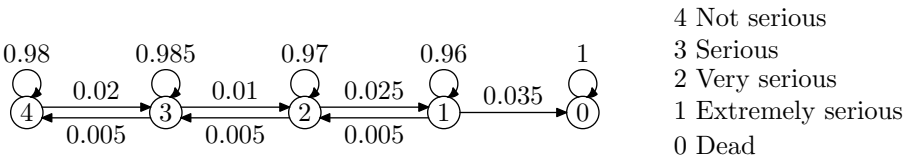

Fig. 11. Markov chain representing transitions between health states for a victim agent in a normal cell, with no intervention from rescuer agents, in SimGenis [Saoud et al. 2006].

Equation 1: the 'health level' of a civilian, which was used as part of the objective in RoboCup Rescue. A 'health level' parameter is perhaps the most common type of parameter in agents which represent human individuals. This is because it is needed for determining whether such an agent is dead or alive, and, as can be seen from Table I, the 'number of fatalities' is one of the most common types of objectives used when optimizing large-scale emergency response[26].

Conceptually, the health parameter may be discrete-valued, continuous, or vector-valued. When the parameter representing health is discrete, Markov chains, and the closely related finite-state machines, are a popular way to model their evolution. In SimGenis for example, a discrete-valued health level parameter is the *only* parameter in the state of agents which represent victims. Figure 11 shows the five discrete values this health parameter can take, and how these evolve in the form of a Markov chain. The transition probabilities between states depend on:

(1) *The victim agent's immediate environment.* In this case, the 'type' of the cell ('normal' or 'danger') that the victim is in. (Recall from Section 3.2, that the environment is modelled as a grid in SimGenis.)

(2) *Whether or not the victim agent is receiving treatment from rescuer agents.* In SimGenis, 'rescuer' agents are used to represent doctors, nurses, fire-fighters and 'rescue chiefs' (supervisors). A doctor may intervene with a victim by:
   (a) *stabilizing* the victim: this increases the probability of their health staying in the same state, and decreases its probability of worsening; or
   (b) *treating* the victim: this increases the probability of their health improving, and decreases its probability of worsening.

   The behaviour of 'rescuer' agents in SimGenis is discussed in more detail later.

The probabilities given in Figure 11 are those used in a 'normal' cell without intervention from 'rescuer' agents. Heuristics, such as the those in (a) and (b) above, are used to generalize from this reference Markov chain, to estimate the transition probabilities in 'normal' cells with intervention from rescuers, and in 'danger' cells with and without intervention from rescuers. As it is responsible for updating the state, this Markov chain represents the `next` method for victim agents in SimGenis. This takes the 'type' of the cell that the agent is in, and whether 'rescuer' agents are stabilizing/treating it, as parameters. These are obtained from

[26]It is also needed for determining whether or not an individual is infected in a pandemic; 'number of people infected' is the most common objective for such an emergency.

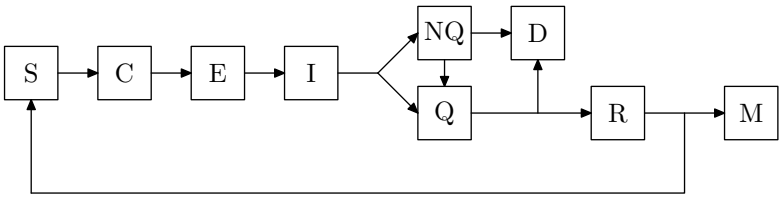

Fig. 12. Evolution between the nine discrete health states in IMAMCR [Khalil et al. 2010].

a `sense` method[27], which is called regularly for each victim agent: in this case, once per second [Saoud et al. 2006][28,29].

When the emergency is a human pandemic, most ABSs use a modified 'SIR' model [Kermack and McKendrick 1927], which also represents the health of human individuals as one of a finite number of discrete states. The original version of SIR has three health states: Susceptible, Infectious and Recovered. In their ABS IMAMCR, Khalil et al. [2010] extended the SIR model to include nine states in total, to model the H1N1 pandemic (swine flu) in Egypt. All agents begin in the *Susceptible* (S) state, i.e. they are prone to becoming infected by the H1N1 virus. Upon coming into direct contact with someone infected with the virus, an agent's health state changes to *in-Contact* (C). With a certain probability, given by a realistic probability distribution, in-contact agents may become infected, in which case their state changes first to *Exposed* (E), then to *Infected* (I). Whilst in the exposed state, agents remain non-contagious. Again, with a certain probability, infected agents may seek medical help, in which case they become *Quarantined* (Q), or not, in which case they remain *Non-Quarantined* (NQ). Non-quarantined agents may later become quarantined, if they later decide to seek medical attention. Both quarantined and non-quarantined have a certain probability of dying (moving to a *Dead* state (D)). Quarantined agents, however, have a probability of becoming *Recovered* (R), in which case they either become *Immune* (M) to the H1N1 virus, or susceptible again. This evolution of health states, which is the `next` method for agents in IMAMCR, is shown in Figure 12.

The health parameter of an agent does not have to be discrete. In the simulated Sarin attack in PLAN-C, for example, it is modelled as a continuous[30] parameter, $h_l \in [0, 1]$, where 0 corresponds to dead and 1 corresponds to perfect health. The maximum rate at which an agent's health can deteriorate is determined by the agent's current health level, relative to:

(1) A 'critical' health level, p_2. Below this level, an agent's health may deteriorate

[27]Note that when we talk of `sense, next` and `action` methods in an ABS, we of course mean their equivalents in the actual code, which may be quite different in structure, never mind in the names of methods used.

[28]It should be noted that the numerical values of the transition probabilities in the Markov chain also depend on the amount of simulated time represented by the discrete time-step between state updates, as they have unit 'per unit of simulated time'.

[29]In JADE, the toolkit that SimGenis uses, agent behaviours can be scheduled to be called at regular time intervals.

[30]In the Java (Repast) code, this is modelled as a `double`.

```
if(rand(0,1) < h_l)
   h_l = h_l+ rand(0,t_l + t_u);
else
{
   double z;
   if(h_l > p_1) z = z_max;
   else if(h_l > p_2) z = z_d;
   else z = z_c;
   h_l = h_l− rand(0,(1-t_l)*z);
}
```

(a)

Parameter	Value
p_1	0.5
p_2	0.2
$z_{\max}$	0.000138 per minute
z_{d}	0.000416 per minute
z_{c}	0.000695 per minute
t_u	not reported

(b)

Fig. 13. (a) Evolution of a victim's health parameter in PLAN-C. (b) Parameter values used in Mysore et al. [2006].

at a maximum rate of z_{c}.

(2) A 'dangerous' health level, p_1, where $p_1 > p_2$. Below this level, but above the 'critical' level, an agent's health may deteriorate at a maximum rate of z_{d}.

Above p_1, an agent's health level may deteriorate at a maximum rate of $z_{\max}$. The parameters $p_1, p_2, z_{\max}, z_{\mathrm{c}}, z_{\mathrm{d}}$ are all global, i.e. they do not vary from agent to agent. The evolution of an agent's health parameter also depends on the current level of treatment it is receiving, which, like the health level itself, is modelled as a real number, $t_l \in [0, 1]$. The algorithm for evolving the health state h_l is given in Figure 13(a), where rand(0,x) returns a number in the range [0,x] using a uniform random distribution, and t_u is a parameter which denotes the 'maximum untreated recovery'. This is the `next` method for PLAN-C. The values used for these global parameters in Mysore et al. [2006] are given in Figure 13(b).

Finally, we note that vector-valued quantities may also be used, to assign health levels to different parts of the human body. For example, the 'Injury Severity Score' [Baker et al. 1974] assigns a value to each of six different parts of the human body: the *head and neck*; the *face*; the *thorax*; the *abdomen*; the *extremities*; and *external soft tissue*. Although these values are then aggregated to produce a final 'score', there is no reason not to allow them retain their ipseity as components of a vector, thus avoiding any loss of information. To the best of our knowledge, such a level of detail has yet to be used in ABS for large-scale emergency response.

3.3.2 *Action selection:* `action`. So far we have reviewed implementations of the health parameter in an agent's state, and possible `next` methods for updating it. However, we also know agents perform actions in the environment. For example, we have already seen, from the viewpoint of a victim agent, how the actions of a rescuer agent can influence the evolution of its health parameter. The selection of which action to perform by an agent is handled by its `action` method, the implementation of which may be (from Section 3.1):

(1) an approximation to what occurs in reality; or

(2) a completely new algorithm which bears no resemblance to how actions are selected in reality (taxon U3).

Agents whose `action` methods are of the second variety are invariably *homo economicus* [Hare and Deadman 2004], that is, they act *rationally*. This is in contrast to *homo sapiens*, who are said to have a 'bounded' rationality [Simon 1957; Rubinstein 1998].

Occasionally however, agents which are intended to be bounded rational make use of an algorithm indicative of perfectly rational behaviour. Consider for example, a `move` action, which results from an `action` method which contains some algorithmic description of movement behaviour. In this case, the behaviour is responsible for determining where the agent is to go, i.e. *path-planning*. For the traversal between two nodes on a graph, popular path-planning algorithms include:

(1) *Dijkstra's algorithm* [Dijkstra 1959], which calculates the *shortest path* between two nodes in a graph (with edges with non-negative weights).
(2) *A* algorithm* [Hart and Raphael 1968], which uses heuristics to achieve efficiency gains over Dijkstra's algorithm.
(3) *D* algorithm* [Stentz 1995], which is similar to the A* algorithm, except edge weights vary in time.
(4) *'longroads' algorithm* [Kleiner et al. 2006], which simplifies the road network before applying Dijkstra's algorithm. It is popular among the RoboCup Rescue community.

These algorithms each construct and optimize an appropriate objective function, to determine the optimal path from a source to a destination. Even should a human travel along the shortest route when making a journey, it is not because they have mentally performed these calculations. Despite this, these algorithms are used in ABSs for agents which are supposed to approximate reality. For example, in SimGenis[31], Saoud et al. [2006] uses Dijkstra's algorithm to navigate ambulances in the environment, but admits '*This assumes that the ambulance driver knows perfectly the whole city ... In future work, we would consider dynamic routing, taking into account unknown areas ...*'. Some ABSs amend algorithms such as those above to incorporate bounded rationality into agent movement. For example, Mysore et al. [2006] achieves it in PLAN-C by introducing a characteristic of panic-like behaviour into the rational path-finding algorithm [Korf 1990] used by civilian agents affected by the Sarin attack.

Considering the action of movement from a programming perspective, any `move` action performed by an agent depends on that agent's movement behaviour, as implemented in its `action` method. Different movement behaviour would result in different `move` actions. Observing then that:

(1) movement behaviour may vary between agents; and
(2) it is good programming practice to 'encapsulate what varies' [Freeman et al. 2004],

it seems a sensible option to represent each distinct movement behaviour in its own class. Furthermore, if[32]:

[31] Which '*mimics the observed team*' [Saoud et al. 2006].
[32] The following terminology is based on C++, but, with the exceptions in parentheses, may be applied to Java.

Cell-Phone Behaviours	
`NullActivity`	No cell-phone activity
`AlwaysCall`	Always make a call
`DistributionBased`	Make a call with a probability based on empirical data
Movement Behaviours	
`NullMovement`	No movement
`RandomMovement`	Random movement
`MoveAndReturnMovement`	Simulates a return journey between the agent's home and workplace
`FleeMovement`	Moves the agent in a straight line away from danger
`BoundedFleeMovement`	As above, but stops moving the agent once it reaches a threshold distance away from the danger
`RoadFleeMovement`	Constrains agents in vehicles to the road network
`CongestionFleeMovement`	Simulates traffic jams
`MixedFleeMovement`	Combines `FleeMovement` and `CongestionFleeMovement`

Table III. Agent Behaviours in WIPER.

(1) each concrete movement class derives from the same abstract base class, overriding a pure virtual `move` method; and

(2) each agent stores a pointer (in Java, reference) to this abstract base class; and

(3) this pointer is set to point to some concrete class at run-time (in Java, the reference is set to a concrete class); and

(4) `action` calls the `move` method using this pointer (in Java, reference),

then the code becomes particularly elegant. This is, in fact, a description of the Strategy design pattern[33] [Gamma et al. 1994], and is exactly how movements are implemented in WIPER [Schoenharl et al. 2009]. The abstract base class for movement behaviours is `MovementModel`. This contains a public pure virtual[34] method `move` which eight concrete behaviour classes then override. A second behaviour, relating to cell-phone use, is implemented similarly. Its abstract base class is `ActivityModel`. This contains a public pure virtual method `checkCall`, which determines if the agent should make a call, and if it should, does. The eight concrete movement behaviours, and three concrete cell-phone behaviours are listed in Table III.

Figure 14 illustrates this design in the form of a (simplified) UML class diagram for an agent in WIPER [Schoenharl 2007]. A `step` method in the `WiperAgent` class (this is the agent's `action` method) is scheduled to be called every time-step (one time-step represents one minute simulated time) using Java annotations in Repast. This calls the `move` and `checkCall` methods on the references to `MovementModel` and `ActivityModel` respectively. As these are set to the appropriate concrete behaviours during the agent construction, they execute the appropriate `move` and `checkCall` actions.

The use of the Strategy design pattern allows the programmer to vary the behaviour(s) in an agent's `action` method, at run-time. If behaviours are known

[33] A more conventional alternative in Java is to use references to an interface which implements a `move` method.

[34] Being implemented in Java (Repast), every non-static public method is virtual by default.

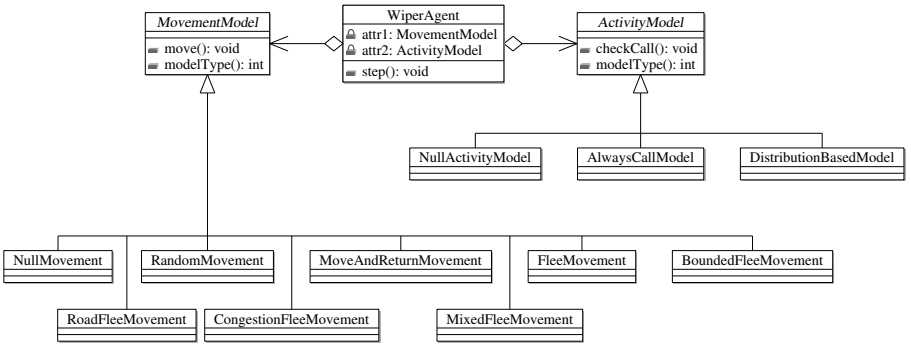

Fig. 14. UML class diagram for `WiperAgent`, showing the use of the Strategy design pattern.

at compile-time, and do not need to be changed at run-time, then the compile-time equivalent, 'policy-based class design' [Alexandrescu 2001] could be used to structure code instead (in C++ at least). With the exception of the incomplete 'meta-agent' project [de Halleux 2003] however[35], as far as we are aware, there are no examples of this being used for the construction of *any* ABS, whatever the application.

Of course, movement is only one example of an agent action, albeit the most common one as it applies to all agents which represent human individuals. Specialized actions exist for specialized types of agent. For example, recall that rescuer agents in SimGenis are used to represent doctors, nurses, fire-fighters and 'rescue chiefs' (supervisors). Each of these 'types' of agent has actions available only to them. From a programming perspective, the UML class diagram for SimGenis agents in Figure 15 shows that each type of rescuer agent is represented by its own class, which derives from an abstract `Rescuer` base class. `Rescuer` and `Victim` agents both derive from a common `Actor` base class. As the JADE toolkit is used, we can infer `Actor` derives from its `Agent` base class (not shown), which itself implements the `Runnable` interface. Such an object-oriented hierarchy of classes is a common way to implement agents in an ABS. By making `action` a virtual function, different behaviours may be implemented in each 'type' of agent class.

The behaviour of the rescuer agents in SimGenis are implemented using heuristic rules, which relate to the following six activities:

(1) The search for victims in the affected area.

(2) The on-site treatment administered by doctors to victims.

(3) The transfer of victims to an advanced medical post (AMP).

(4) The transfer of victims from an AMP to a hospital.

(5) The routing of ambulances to hospital.

(6) The overall management of the rescue process.

[35]The fact that "*using policies, creating agents with different dynamics and behaviors is as simple as changing some templates parameters*" is cited as the major idea behind the project.

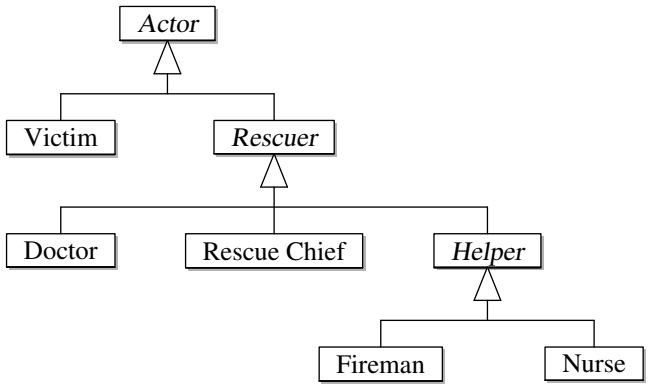

Fig. 15. UML class diagram for agents in SimGenis [Saoud et al. 2004].

Search for victims	
E1	Explorers move randomly; only fire-fighters can pass through 'danger' cell.
E2	Explorers move in an organized way, without visiting same cell more than once.
On-site treatment	
F1	Doctors look for nearest victim within their vision.
F2	If doctor has already visited the site, they go towards nearest already known victim.
F3	If doctor has already visited the site, they go towards most seriously injured already known victim.
A1	Doctors examine victim by questioning and observation.
A2	Doctors examine victim based on reading victim's health-parameter history (paper medical form).
A3	As A2, but electronic medical form.
T	Victims in state 1 or 2 need stabilizing first, or on-site treatment, or both, and receive higher evacuation priority than victims in state 3 or 4.

Table IV. Examples of the heuristic rules used to implement rescuer behaviours in SimGenis. Key to rule labels: E: 'exploration' rule; F: 'finding victim' rule; A: 'assessment of victim' rule; T: 'treatment of victim' rule.

The rules relating to the search for victims, and to on-site treatment, are shown in Table IV. These, and the rules for the other four activities, are used to implement the `action` methods in the `Doctor, Fireman, Nurse` and `RescueChief` classes. Clearly, in order to use these rules, rescuer agents implement a `sense` method, to give them perception. Although no details are given on how `sense` is implemented, we are told that rescuer agents can distinguish between victims and other rescuer agents, and between the three different types of cell. Furthermore, each rescuer agent's state contains a 'visibility' parameter, which determines how far they can see. Other parameters in the state of a rescuer agent include its location, speed, and a parameter representing the action which they are currently performing.

From the UML class diagram in Figure 15, we can see that agents are used to represent only human individuals in SimGenis. Sometimes however, it is natural to think of non-human entities as making decisions, e.g. we may talk of a hospital 'deciding' to release a patient, or of an ambulance 'deciding' to travel a particular

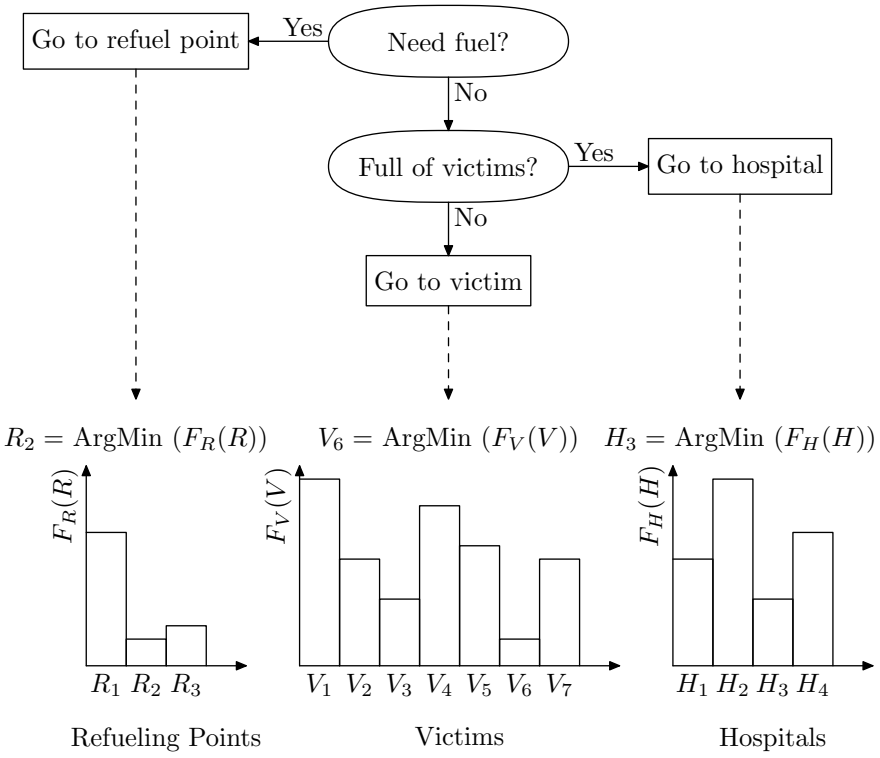

Fig. 16. Two-step `action` method for an ambulance agent in AROUND [Chu et al. 2009].

route to an incident. Because of this, many ABSs model non-human decision-making entities as agents, a process referred to as *agentification* by Gaumé et al. [1999]. Hare and Deadman [2004] assert agentification "*brings some reality into the discussion of agents in simulation in that it makes clear that agents should be designed to fit simulation needs*".

For example, in AROUND [Chu et al. 2009], *ambulances* are represented as agents. The behaviour of an ambulance agent, as implemented in its `action` method, is modelled as a two step process:

(1) In the first step, a *decision tree* is used to decide what *type* of action to perform.
(2) In the second step, a *utility function* is used to determine the finer details of this action.

This is illustrated in Figure 16. The set of actions available to the ambulance is $Ac = \{$ 'go to refuel point', 'go to hospital', 'go to victim'$\}$. The decision tree is used to choose which of these three actions to execute. If it is decided to go to a refuel point, a utility function F_R determines the precise refueling point R to visit (R_1,R_2 or R_3), with the point minimizing F_R being selected (in this case R_2). Similarly, for the other two types of action, utility functions F_H and F_V are used to choose exactly which hospital to visit and which victim to collect, respectively.

The utility functions take the form of a weighted sum of different objectives. The simulation begins with a set of default weights. However, experts (paramedics) interact with the AROUND simulation as it runs, and can overrule what ambulance agents decide to do. When this happens, the weights are optimized, to yield an objective which is most consistent with the expert's decision making.

RoboCup Rescue also makes use of agentification. Recall from Figure 10 that it used a centralized kernel to update its virtual environment from a set of simulators. Also connected to this kernel was an *agents* component. It contains two main categories of agents:

(1) *platoon agents*: 'fire brigades', 'police forces' and 'ambulances'; and
(2) *centre agents*: 'fire stations', 'police offices' and 'ambulance centres'. These send and receive messages to and from their corresponding platoons, in order to aid their coordination.

Users of RoboCup Rescue must implement the `actions` for each of these agent types. From the 2010 team source code [Robocup Rescue Simulation Project 2010], there are two main routes to doing this:

(1) One starting point is the abstract base class `StandardAgent`, from which two separate base classes, one for platoon agents and another for centre agents, may be derived. This approach is shown in the UML class diagram in Figure 17, where `AbstractPlatoonAgent` and `CenterAgent` are the base classes for platoon agents and centre agents respectively, and was the one taken by the 2010 competition winners, RoboAKUT [Akin et al. 2010].
(2) Alternatively, users can derive their agents directly from `StandardAgent`'s base class, `AbstractAgent`, as IAMRescue [Fave et al. 2010], the runners-up of the 2010 competition, did.

Either way, the platoon and centre agents each derive from `AbstractAgent` (which, for completeness, derives from `AbstractComponent`). `AbstractAgent` contains a function `processSense`, which is called at regular intervals by the kernel. This is the `sense` method for each agent. This takes as an argument an object of type `KASense` which contains perception information for the agent[36]. From the source code, we note that `processSense` makes use of the Template design pattern [Gamma et al. 1994], with the 'hook' being a pure virtual function `think` which each subclass of `AbstractAgent` must implement. The function `think` takes as parameters:

(1) the set of changes which the agent has observed since the previous timestep,
(2) the set of communication messages that the agent hears, and
(3) the timestep.

Using this sensory information, each type of agent selects an action to perform, and sends a message back to the kernel informing it of its choice. Thus `think` represents the `action` method for RoboCup Rescue agents.

[36]This is determined by the kernel, which uses the R-tree spatial index to determine the entities in 'domain objects' which are in the line-of-sight of the agent.

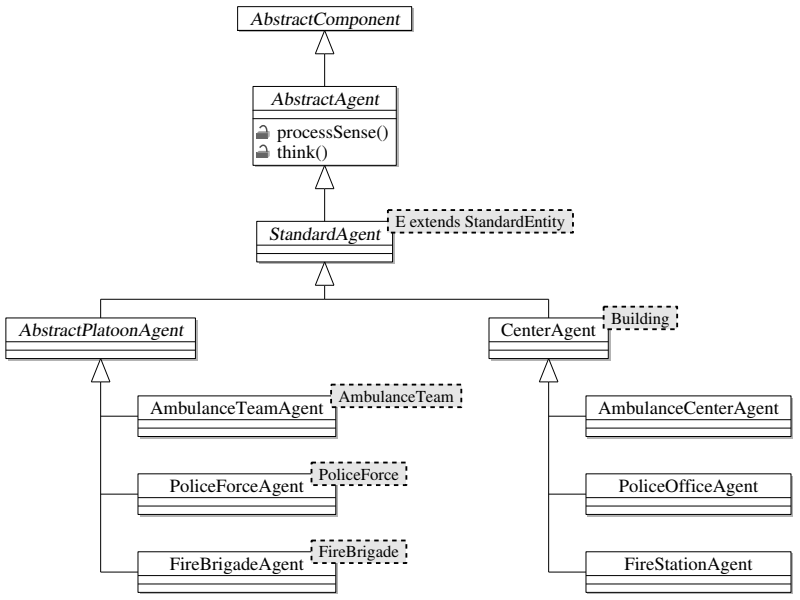

Fig. 17. UML class diagram for agents developed for RoboCup Rescue.

The set of actions, Ac, from which an action is selected, depends on the type of agent. The actions for the three different types of platoon agent are shown in Table V. Some of these actions pertain to communication, which is designed to be imperfect (as messages drop). It is the responsibility of the user of RoboCup Rescue to implement, in `think`, the action selection algorithms for the platoon agents and their centres, in order to optimize the performance measure, given in Equation 1.

The ambulance agent in AROUND was an example of a *learning* agent. Its behaviour adapts with time, in response to feedback from the experts interacting with the simulation. An example of a learning agent in RoboCup Rescue is found in Runka [2010]. There, each different type of platoon agent[39] in RoboCup Rescue is modeled as a tree structure, which represents a decision tree. Optimal tree structures are sought using genetic programming [Koza 1992]. Two approaches are taken:

(1) In the first approach, three separate tree structures, each representing one of the platoon types of agent in RoboCup Rescue, make up a single 'individual' in the genetic program. The score of an individual is calculated by running a RoboCup Rescue simulation, using its three tree structures as platoon agents. A population of individuals is created, and the score of each individual calculated. Individuals are selected for crossover (with a probability which increases

[37] Using the `StandardCommunicationModel`.
[38] Using the `ChannelCommunicationModel`.
[39] A 'distributed' approach is taken, whereby the responsibility for decision making lies entirely with the platoon agents. A 'centralized' approach, on the other hand, requires that the platoon agents send all their information to their centers, which then issue them with instructions.

All	
`sendRest`	Do nothing
`sendMove`	Move agent
`sendSay`	Communicate to other nearby agents[37]
`sendTell`	Communicate via transmission[37]
`sendSpeak`	Communicate by voice or radio[38]
`sendSubscribe`	Subscribe to a set of communication channels
PoliceForce	
`sendClear`	Clear a road
FireBrigade	
`sendExtinguish`	Extinguish a building
AmbulanceTeam	
`sendRescue`	Remove a trapped civilian from a building
`sendLoad`	Load an injured civilian for transport
`sendUnload`	Unload a carried civilian

Table V. Platoon agent actions in RoboCup Rescue.

with their score), creating a new population of individuals with favourable tree substructures. Populations evolve for several generations, in the search for the optimal individual.

(2) In the second approach, an individual represents only one type of platoon agent. Three separate populations thus exist, which evolve separately. In this case, the score of an individual is related only to the tasks that type of agent is responsible for: thus the ambulance team score is based only on the health points at the end of the simulation, the fire brigade score is dependent only on the area of unburnt buildings, and the police force score is dependent only on the number of cleared blockages.

An example fire brigade agent is given in Figure 18. Its tree-structure may be interpreted as follows: 'if the agent is at a fire, then put out all the fires at the nearest building; otherwise, move to the nearest building on fire'.

This type of agent is quite appealing. In particular, it does not need its behaviour to be manually coded (other than the initial genetic programming code), nor does it require a human expert to learn from, as AROUND agents did.

Whilst the agents in AROUND learnt 'real' behaviour from the experts they represented, these genetically programmed agents learn 'optimal' behaviour, as the fitness function driving their evolution is the objective function in Equation 1. Initial results using them, however, were disappointing [Runka 2010], and further work is required before if they are to become competitive with manually written action selection mechanisms.

3.4 Scale

For an ABS to model *large-scale* emergency response, either the number of agents representing displaced, injured or killed human individuals is greater than one hundred, *or* the virtual environment in which the agents are situated represents a geographical area greater than ten square kilometres, *or* both. There are two reasons why using a single computing core to run such ABSs may be inadequate [Hager and Wellein 2011]:

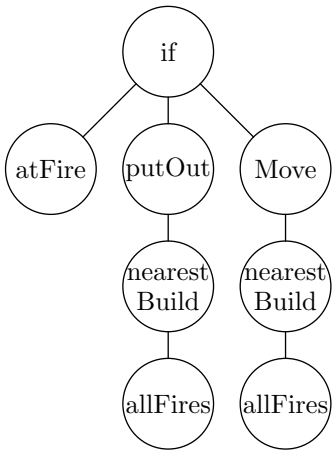

Fig. 18. An example tree structure representation of a fire brigade agent for RoboCup Rescue, built using genetic programming in [Runka 2010].

(1) *A single core is too slow to perform the simulation in a tolerable amount of time.* For example, Schoenharl et al. [2009] shows how the run-time of WIPER increases as the number of agents increases (linearly until approximately 200,000 agents, non-linearly thereafter). This places an upper limit on the number of agents which can be simulated in real-time.

(2) *The memory requirements of the ABS exceed the amount of main memory available on a single machine.* For example, RoboCup Rescue was designed to model the 1995 Kobe earthquake, in which over 6,000 people died, and over 300,000 people were made homeless. However, RoboCup Rescue typically only models one hundred or so agents. In an investigation of its scalability, Sarika [2010] reports "*the number of rescue agents was increased to 175. However, the simulation failed, the reason being either the simulators or the agent program ran out of memory during the initialization process*".

Going beyond the use of one computing core is parallelization, and requires a parallel computer. There are three main classes of parallel computer[40] which are in use today [Fujimoto 2000]:

(1) *Shared memory multiprocessors*, which allow multiple processors to work on a common, shared physical address space.

(2) *Distributed memory multicomputers*, each processor of which has its own exclusive memory to which no other processor has direct access.

(3) *Single Instruction Multiple Data (SIMD) machines*, each processor of which executes the same instruction on different data during program execution.

[40]As well as parallel computers, there are distributed computers [Fujimoto 2000]. The difference between a parallel computer and a distributed computer is the physical area they occupy: the processors of a parallel computer are in close proximity (a single machine, or a cabinet), whilst the processors of a distributed computer can span anything from a building to the entire planet.

3.4.1 *Shared memory parallelism.* Multi-core computers are now commonplace, and so exploiting shared memory parallelism is an issue facing all software [Sutter 2005; Sutter and Larus 2005], not just ABS. The most common way to exploit it in an ABS is to enable agents to act concurrently. Fortunately, the most common programming languages provide support for concurrent programming. Examples of the use of shared memory parallelism include:

Java. Parker [2007] exploited shared (and distributed) memory parallelism in an agent-based epidemic model, using Java. On his choice of programming language he states: "*Java is an excellent language with which to develop large agent based models. Java provides many built in features like RMI, serialization, and thread management tools that allow a distributed model to be easily developed.*"

ABSs which are built using the Java-based toolkit JADE take advantage of shared memory parallelism, as each agent derives from the `agent` class which implements the `runnable` interface and thus executes in a Java thread. The Java Virtual Machine allows processes to run multiple threads concurrently.

C/C++. In an investigation into strategies on how to mitigate an influenza pandemic in the U.S. and in the U.K., Ferguson et al. [2003] describe an ABS written in C, which used the OpenMP libraries [Massaioli et al. 2005] to exploit shared memory parallelism. This was necessary, as the U.S. simulation used 55 GB of RAM.

3.4.2 *Distributed memory parallelism.* IDSS (Integrated Disaster mitigation Simulation System) [Koto and Takeuchi 2003] is one of the earliest ABSs for large-scale emergency response which attempts to use distributed memory parallelism. It uses RoboCup Rescue's architecture as a starting point for the design of a new ABS which uses distributed memory parallelism. In IDSS, a large virtual environment is first split into smaller sub-regions. Each one of these sub-regions is then simulated on a separate computer. The attributes of the entities in each sub-region, referred to as 'state variables', are stored in a table called a 'space-time graph'. Each cell of a space-time graph gives the value of a state variable, determined by the row of the table, at a particular instance in time, determined by the column of the table.

On each computer, a local 'kernel' program is responsible for the management of the space-time graph. Communicating with each kernel is a set of sub-simulator programs, also running locally. Each sub-simulator handles a different dynamic aspect of the simulation, such as fire-spread, or building collapse, just as in RoboCup Rescue. Each kernel communicates with the other kernels which represent its neighbouring sub-regions, using a 100MB Ethernet interconnect. The connection of two kernels, each of which has multiple sub-simulators, is shown in Figure 19. In [Koto and Takeuchi 2003], a 4.5 km^2 region of Kobe city, Japan, was simulated on a 34 PC cluster. The performances of traffic and fire sub-simulators were shown to increase linearly as the number of computers used increased; however, at this point the IDSS did not contain any agents.

Agents were introduced to IDSS by Takeuchi [2005]. Each agent communicates with the kernel via a special sub-simulator called an Agent Proxy, labelled as APX in Figure 19. One APX sub-simulator can contain multiple agents, and a kernel can communicate with multiple APX sub-simulators. One APX (and the agents it handles) runs in one single process. This approach reduces the number of agent-kernel

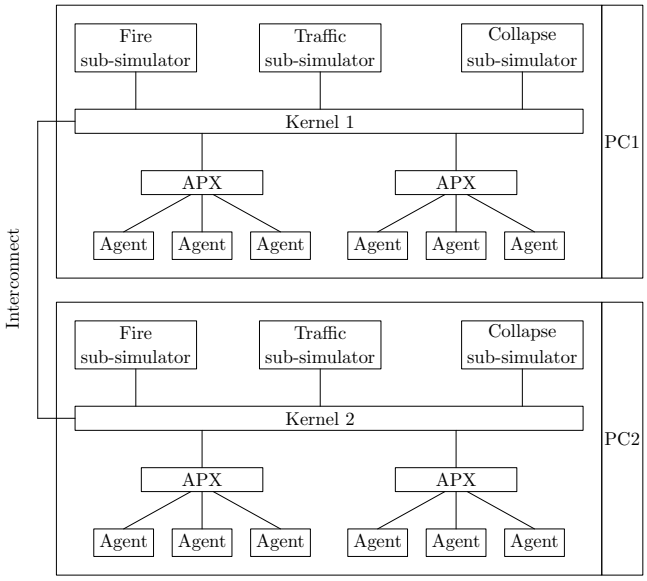

Fig. 19. Architecture of IDSS, running on two PCs.

messages, as each APX bundles together the messages for all the agents it contains, before sending it to the kernel. Being situated in a distributed environment, the issues of an agent migrating from one APX to another, and communication between agents in different APXs are noted by Takeuchi [2005]. Just as with the environment, scale is the prime consideration. A figure of 10,000 agents is given as a minimum target, however it is unreported if this was actually realized.

Another ABS for large-scale emergency response which uses distributed memory parallelism is EpiSimS. In [Valle et al. 2006], it was used to simulate the daily routines of 16,106,535 individuals in Los Angeles, which was modelled as a graph consisting of 562,452 locations. These locations were distributed over 106 processors, each with 2GB of local memory. Agents representing individuals were able to pass between processors (if their daily routine took them from a location on one processor to a location on another). Synchronisation was enforced through use of a *master-worker* scheme: after each time step, each of the (worker) processors simulating locations communicated back to a single master process, which ensured no processes started got a certain amount of time ('*slack time*') ahead of any other.

Implemented at Los Alamos National Laboratory during the mid 1990s, EpiSimS was built using a distributed computing toolkit, which was also used for implementing the transport simulation TRANSIMS [Smith et al. 1995]. In 2009, that toolkit was redesigned and updated, to become ABM++ [Roberts 2010], which is C++ based and makes use of MPI. The appearance of this toolkit, and the similar Repast HPC [Collier and North 2011], will hopefully ease the development of future distributed memory ABSs for large-scale emergency response.

3.4.3 *SIMD parallelism.* The use of Graphical Processing Units (GPUs) for general purpose computing has become increasingly popular in recent years, especially

since the introduction of CUDA (Compute Unified Device Architecture) by NVIDIA in early 2007 [Kirk and Hwu 2010]. GPUs belong to the class of Single Instruction Multiple Data (SIMD) parallel computers: they can process the same instructions on multiple data concurrently. Sethia and Karlapalem [2010] argue this is well suited to agent-based simulation, as:

(1) each type of agent is represented by the same code (single instruction), and
(2) there exists multiple instances of each type of agent (multiple data).

They outline its potential application, using CUDA, to RoboCup Rescue. A major bottleneck in RoboCup Rescue occurs in the kernel, where perceptions for each agent are computed in serial. Therefore, the calculation of percepts for the agents concurrently on a GPU is proposed.

RoboCup Rescue is written in Java, whilst CUDA requires code written in C/C++. Therefore this proposed use of GPUs requires a rewrite of much code, in particular:

(1) the calculation of percepts in the kernel,
(2) the simulator code, and
(3) agent action selection code.

Instead, Sethia and Karlapalem [2010] emulate the use of a GPU in RoboCup Rescue, and so their testing can only really be viewed as proof-of-principle.

Although it is probably only a matter of time before the use of GPU computing becomes more common in ABS for large-scale emergency response, perhaps the biggest barrier at present is that experienced here: CUDA, the most popular tool currently for GPU computing, requires C/C++, however most popular agent-based toolkits for building ABS are Java-based. For example, Nikolai and Madey [2009a] state that 42% of agent-based toolkits are Java-based. However, as these include some very popular toolkits such as JADE and Repast, the actual percentage of *ABSs using* a Java-based toolkit is likely to be much higher.

4. SUMMARY AND FUTURE RESEARCH

4.1 Summary

ABS for large-scale emergency response has been reviewed from four perspectives: *usage*; implementation of the *environment*; implementation of the *agents*; and *scalability*.

We have seen that ABS may be used either during *preparedness*, or *during the real-time response*. The behaviour of agents is based on the behaviour of the real world decision-making entities they represent, unless the purpose of the research is to design behaviours *ex novo*: the search for 'smart new algorithms' as Carley et al. [2006] puts it. As one single execution of an ABS simulates one course of action to one particular scenario, multiple executions are needed to determine the description of the optimal response. How this description depends on the characteristics of the emergency itself is often investigated using a traditional design of experiments.

The *environment* is most commonly implemented either as a grid, or using a vector GIS data source. Using vector GIS data has the advantage that entities of arbitrary shape can be modelled accurately. Furthermore, transport networks

may be modelled using graphs. The level of detail in the virtual environment influences the results of the simulation, as seen in the work of Sato and Takahashi [2011], however with increased level of detail comes increased computational burden. *Dynamic* aspects of the environment may be modelled using separate simulators which communicate with the ABS, or alternatively they may be modelled using agents within the ABS itself.

Agents are used to represent human individuals, such as civilians and emergency responders. Some non-human entities are also modelled as agents, if they can be viewed as making decisions: examples include hospitals and ambulances. Object oriented hierarchies are typically used to create different 'types' of agent, whilst different agent behaviours may be encapsulated using the Strategy design pattern. Heuristic rules, finite state-machines, utility functions and decision trees have all been used as the basis for implementing decision-making. The state of an agent may contain multiple parameters which influence its behaviour. One particular parameter additionally often plays a part in the objective of the response, namely the health level of human individuals; this is usually modelled as evolving stochastically.

Achieving the *large-scales* involved, either in terms of numbers of agents, or in terms of the size of the environment, often requires the use of parallelism. Distributed memory parallelism is useful for modelling large geographical areas, but may not be an option when using some agent-based toolkits. More common is the use of shared memory parallelism, which enables the concurrent execution of agent actions. GPU computing is also an option, and (as of 2010) is just beginning to be applied to large-scale emergency response.

4.2 Future research

4.2.1 *Usage.* The potential for ABS to contribute to policy formulation in emergency response, where commonly held beliefs have been cited as lacking in supporting evidence [der Heide 2006; Bledsoe 2007], should not be overlooked.

In the case of the 'golden hour', the popular notion that trauma casualties will have "*better outcomes if they are provided definitive care within 60 minutes of the occurrence of their injuries*", an investigation into the origin of the term [Lerner and Moscati 2001] found nothing to suggest it had been derived from evidence. Nevertheless, it has gone on to become a key assumption used in emergency response practice, despite statistical analyses [Osterwalder 2002; Newgard et al. 2010] concluding that the hypothesis should be rejected. A related assumption states that the response time, the time taken by the Ambulance Service to arrive on scene, is highly influential on the resulting mortality - as a result, targets on response times are typically imposed in an effort to improve outcomes [LAS 2011]. Again, data to support this claim is lacking, with Turner et al. [2006] noting that "*overall, there is little evidence in the data that faster response times have led to better outcomes*" and similar findings reported by Petri et al. [1995] and Pons and Markovchick [2002]. The potential contribution of ABS to these debates could entail not only the supply of further data to help assess the validity of the assumptions in question, but also the ability to predict the impact of their removal in terms of risk and benefit.

As a result of the perceived importance of speed in the response operation, lights and sirens are commonly used by the Ambulance Service in an effort to decrease travel times. It has been shown, however [Hunt et al. 1995], that the benefit of such

measures are minimal in practice, decreasing response time by only 43.5 seconds on average. Furthermore, despite this limited gain in transport time, a large proportion (39.4%) of cases where lights and sirens were in use were noted by [Lacher and Bausher 1997] to involve casualties who were in a stable condition. The authors go on to call for "*additional studies to evaluate the effect of lights and sirens on transport time and patient outcome*" in order to "*aid the development of standardized protocols for their appropriate use*", a request which the use of ABS could go some way to fulfilling - in particular, work comparing the benefit of lights and sirens to the associated risk to both ambulance personnel and their charge [Maguire et al. 2002] would be valuable. Similarly, the use of helicopters as means to transport casualties is by no means a risk-free operation, with the occupational death rate of air medical crew far exceeding the average across all working citizens [Bledsoe 2007]. As with the use of lights and sirens in ground transportation, research has suggested that use of medical helicopters may not result in any benefit to patients [Bledsoe et al. 2006] and as such the continued investment in them should be reviewed, a task to which ABS could make a valuable contribution.

4.2.2 *Implementation.* Regarding the implementation of ABS, work is beginning to appear comparing how different representations of the environment influence large-scale emergency response simulations [Sato and Takahashi 2011]. Further work like this, covering more representations, and for different scenarios, would be of great benefit to future ABS developers. Similar work using different agent representations would also be useful, and along with general advice such as [Müller 1999], would go towards identifying which representations lie inside the 'Medawar' zone for different types of large-scale emergency.

Although the simulation of large-scale emergency response is clearly an area that could benefit from high performance computing, most ABSs in this domain have been single-process. We expect that the appearance of toolkits for distributed agent-based simulation, such as Repast HPC [Collier and North 2011], will go a long way towards improving this situation. Indeed, as projects become ever more ambitious [Helbing et al. 2010], even an ABS for planet-sized emergencies, in which there as many agents as there are people on Earth, seems to be an increasingly realistic goal.

REFERENCES

Akin, H. L., Yilmaz, O., and Sevim, M. M. 2010. RoboAKUT 2010 Rescue Simulation League Agent Team Description. http://roborescue.sourceforge.net/2010/tdps/agents/roboakut.pdf.

Alexandrescu, A. 2001. *Modern C++ Design: Generic Programming and Design Patterns Applied*. Addison Wesley.

Allan, R. J. 2010. Survey of Agent Based Modelling and Simulation Tools. Tech. Rep. DL-TR-2010-007, Computational Science and Engineering Department, STFC Daresbury Laboratory.

Amouroux, E., Chu, T., Boucher, A., and Drogoul, A. 2009. GAMA: An Environment for Implementing and Running Spatially Explicit Multi-agent Simulations. *Lecture Notes In Artificial Intelligence 5044*, 359–371.

An, V. D. 2008. Implantation des protocoles de communication FIPA dans la plate-forme GAMA. M.S. thesis, L'Institut de la Francophonie pour l'Informatique.

Antoniotti, M., Park, F., Policriti, A., Ugel, N., and Mishra, B. 2003. Foundations of a query and simulation system for the modeling of biochemical and biological processes. In *Proc. of the Pacific Symposium of Biocomputing (PSB03)*. Vol. 58. 59.

ArcGIS. 2010. ArcGIS: A Complete Integrated System. http://www.esri.com/software/arcgis/index.html.

Au-Yeung, S. W. M., Harrison, P. G., and Knottenbelt, W. J. 2006. A queueing network model of patient flow in an accident and emergency department. In *Proc. 20th Annual European and Simulation Modelling Conference*. 60–67.

Australian Government Department of Defence. 2005. Simulation verification, validation and accreditation guide. Tech. rep., Australian Defence Simulation Office Department of Defence, Canberra.

Axtell, R., Axelrod, R., Epstein, J., and Cohen, M. 1996. Aligning simulation models: A case study and results. *Computational & Mathematical Organization Theory 1,* 2, 123–141.

Baker, S., O'Neill, B., Jr, W. H., and Long, W. 1974. The Injury Severity Score: a method for describing patients with multiple injuries and evaluating emergency care. *The Journal of Trauma 14,* 3, 187–196.

Balci, O. 1997. Verification, validation and accreditation of simulation models. In *Proceedings of the 1997 Winter Simulation Conference*, S. Andradóttir, K. J. Healy, D. H. Withers, and B. L. Nelson, Eds. 135–141.

Barrett, C. L., Eubank, S. G., and Smith, J. P. 2005. If Smallpox Strikes Portland ... *Scientific American 292,* 3, 42–49.

BBC. 2009. Egypt slaughters pigs to stop flu. http://news.bbc.co.uk/1/hi/8024946.stm.

Beck, K. and Andres, C. 2004. *Extreme Programming Explained: Embrace Change*. Addison Wesley.

Berren, M. R., Beigel, A., and Ghertner, S. 1980. A typology for the classification of disasters. *Community Mental Health Journal 16,* 2, 103–111.

Bharath, G. K. and Silverman, B. G. 2010. Validating agent based social systems models. In *Proceedings of the 2010 Winter Simulation Conference*, B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesa, Eds.

Bledsoe, B., Wesley, A., Eckstein, M., Dunn, T., and O'Keefe, M. 2006. Helicopter scene transport of trauma patients with nonlife-threatening injuries: a meta-analysis. *The Journal of trauma 60,* 6, 1257.

Bledsoe, B. E. 2007. Myths of Modern EMS. http://www.bryanbledsoe.com/data/pdf/handouts/powerpoint/Myths of Modern EMS (Revised).ppt.

Bonabeau, E. 2002. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences 99*, 7280–7287.

Boucher, A., Canal, R., Chu, T., Drogoul, A., Gaudou, B., Le, V., Moraru, V., Van Nguyen, N., Vu, Q., Taillandier, P., et al. 2009. The around project: Adapting robotic disaster response to developing countries. In *Safety, Security & Rescue Robotics (SSRR), 2009 IEEE International Workshop on*. IEEE, 1–6.

Branke, J., Deb, K., Dierolf, H., and Osswald, M. 2004. Finding knees in multi-objective optimization. In *Parallel Problem Solving from Nature-PPSN VIII*. Springer, 722–731.

Bratman, M. E., Isreal, D. J., and Pollack, M. E. 1988. Plans and resource-bounded practical reasoning. *Computational Intelligence 4*, 349–355.

Brenner, M., Wijermans, N., Nussle, T., and De Boer, B. 2005. Simulating and controlling civilian crowds in robocup rescue. In *Proceedings of Robocup 2005*.

Brown, D., Riolo, R., Robinson, D., North, M., and Rand, W. 2005. Spatial process and data models: Toward integration of agent-based models and GIS. *Journal of Geographical Systems 7,* 1, 25–47.

Burke, E. and Hendry, C. 1997. Decision making on the london incident ground: an exploratory study. *Journal of Managerial Psychology 12,* 1, 40–47.

Carley, K. M., Altman, N., Kaminsky, B., Nave, D., and Yahja, A. 2004. BioWar: A City-Scale Multi-Agent Network Model of Weaponized Biological Attacks. Tech. Rep. CMU-ISRI-04-101, Carnegie Mellon University.

Carley, K. M., Fridsma, D. B., Casman, E., Yahja, A., Altman, N., Chen, L.-C., Kaminsky, B., and Nave, D. 2006. BioWar: scalable agent-based model of bioattacks. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans 36,* 2 (March), 252–265.

Castelfranchi, C. 2010. Bye-Bye Agents? Not. *IEEE Internet Computing 14,* 2, 93–96.

Castle, C. J. E. and Crooks, A. T. 2006. Principles and concepts of agent-based modelling for developing geospatial simulations. Tech. Rep. Paper 110, Centre for Advanced Spatial Analysis, University College London. September.

Challenger, R., Clegg, C. W., and Robinson, M. A. 2009a. Understanding Crowd Behaviours: Guidance and Lessons Identified. U.K. Cabinet Office.

Challenger, R., Clegg, C. W., and Robinson, M. A. 2009b. Understanding Crowd Behaviours: Simulation Tools. U.K. Cabinet Office.

Chen, F., Zhai, Z., and Madey, G. 2011. Dynamic Adaptive Disaster Simulation: Developing a Predictive Model of Emergency Behavior Using Cell Phones and GIS Data. In *Proceedings of SpringSim 2011*. Boston.

Chen, L., Kaminsky, B., Tummino, T., Carley, K., Casman, E., Fridsma, D., and Yahja, A. 2004. Aligning simulation models of smallpox outbreaks. *Intelligence and Security Informatics*, 1–16.

Chu, T., Drogoul, A., Boucher, A., and Zucker, J. 2009. Interactive Learning of Independent Experts Criteria for Rescue Simulations. *Journal of Universal Computer Science 15,* 13, 2701–2725.

Chu, T.-Q., Boucher, A., Drogoul, A., Vo, D.-A., Nguyen, H.-P., and Zucker, J.-D. 2008. Interactive learning of expert criteria for rescue simulations. *Lecture Notes in Artificial Intelligence 5357*, 127–138.

Collier, N. and North, M. 2011. Repast HPC: A platform for Large-scale Agent-based Modeling. In *Large-Scale Computing Techniques for Complex System Simulations*, W. Dubitzky, K. Kurowski, and B. Schott, Eds. Wiley.

Connelly, L. G. and Bair, A. E. 2004. Discrete event simulation of emergency department activity: A platform for system-level operations research. *Academic Emergency Medicine 11,* 11, 1177–1185.

Crook, P. 2010. Exercise Orion - Practical Planning. *Fire Magazine*, 32–33.

Crooks, A. 2008. Constructing and Implementing an Agent-Based Model of Residential Segregation through Vector GIS. Tech. Rep. 133, Centre for Advanced Spatial Analysis, University College London. April.

Crooks, A., Castle, C., and Batty, M. 2008. Key challenges in agent-based modelling for geospatial simulation. *Computers, Environment and Urban Systems 32,* 6 (November), 417–430.

Crooks, A. T. 2007. The Repast Simulation/Modelling System for Geospatial Simulation. Tech. Rep. Paper 123, Centre for Advanced Spatial Analysis, University College London. September.

da Silva, M. L., Kostakos, V., and Matsumoto, M. 2008. Improving emergency response to mass casualty incidents. In *Proceedings of Sixth Annual IEEE International Conference on Pervasive Computing and Communications*. IEEE Computer Society, Los Alamitos, CA, USA, 256–259.

Darema, F. 2004. Dynamic Data Driven Applications Systems: A New Paradigm for Application Simulations and Measurements. *Lecture Notes in Computer Science 3038*, 662–669.

de Andrade, P. R., Monteiro, A. M. V., and Camara, G. 2008. Entities and relations for agent-based modelling of complex spatial systems. In *Proceedings of the First Brazilian Workshop on Social Simulations*. 52–63.

de Boer, J. 1990. Definition and classification of disasters: Introduction of a disaster severity scale. *The Journal of Emergency Medicine 8*, 591–595.

de Halleux, J. 2003. MetaAgent, a Steering Behavior Template Library. http://www.codeproject.com/KB/library/metaagent.aspx.

Department of Defense. 2009. Dod modeling and simulation (m&s) verification, validation, and accreditation. http://www.dtic.mil/whs/directives/corres/pdf/500061p.pdf.

der Heide, E. A. 2006. The importance of evidence-based disaster planning. *Annals of Emergency Medicine 47,* 1, 34–49.

Dijkstra, E. 1959. A note on two problems in connexion with graphs. *Numerische mathematik 1,* 1, 269–271.

Dombrowsky, W. R. 1995. Again and again: Is a disaster what we call "disaster"? some conceptual notes on conceptualizing the object of disaster sociology. *International Journal of Mass Emergencies and Disasters 13,* 3 (November), 241–254.

Drogoul, A., Vanbergue, D., and Meurisse, T. 2003. Multi-agent based simulation: where are the agents? *Lecture Notes in Computer Science 2581*, 1–15.

Duong, D. 2010. Verification, validation, and accreditation (vv&a) of social simulations. In *Spring Simulation Interoperability Workshop, Orlando*.

Epstein, J. M. 2009. Modelling to contain pandemics. *Nature 460*, 687.

Epstein, J. M. and Axtell, R. 1996. *Growing Artificial Societies*. Brookings Institutuion Press.

ESRI. 1998. ESRI Shapefile Technical Description. http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf.

Eubank, S. 2002. Scalable, efficient epidemiological simulation. In *Proceedings of the 2002 ACM symposium on Applied Computing*. 139–145.

Fave, F. M. D., Packer, H., Pryymak, O., Stein, S., Stranders, R., Tran-Thanh, L., Vytelingum, P., Williamson, S. A., and Jennings, N. R. 2010. Rescue Simulation League Team Description IAMRescue. http://roborescue.sourceforge.net/2010/tdps/agents/iamrescue.pdf.

FEMA. 2007. National preparedness guidelines. http://www.fema.gov/pdf/government/npg.pdf.

FEMA. 2011. Federal emergency management agency. http://www.fema.gov/hazard/types.shtm.

Ferguson, N., Keeling, M., Edmunds, W., Gani, R., Grenfell, B., Anderson, R., and Leach, S. 2003. Planning for smallpox outbreaks. *Nature 425,* 6959, 681–685.

Finkel, R. A. and Bentley, J. L. 1974. Quad trees: a data structure for retrieval on composite keys. *Acta Informatica 4*, 1–9.

Fischer III, H. W. 2003. The Sociology of Disaster: Definitions, Research Questions, Measurements in a Post-September 11, 2001 Environment. In *presented at the annual meeting of the American Sociological Association, Atlanta Hilton Hotel, Atlanta, GA*.

Floyd, R. W. 1962. Algorithm 97: Shortest Path. *Communications of the ACM 5,* 6 (June), 345.

Franklin, S. and Graesser, A. 1997. Is it an agent, or just a program?: A taxonomy for autonomous agents. *Lecture Notes in Computer Science 1193*, 21–35.

Freeman, E. T., Robson, E., Bates, B., and Sierra, K. 2004. *Head First Design Patterns*. O'Reilly Media.

Fujimoto, R. M. 2000. *Parallel and Distributed Simulation Systems*. Wiley.

Gad-El-Hak, M. 2009. The art and science of large-scale disasters. *Bulletin of the Polish Academy of Sciences: Technical Sciences 57,* 1, 3–34.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. 1994. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.

Gaumé, F., Fallet, B., Ferrand, N., and Chastel, J.-M. 1999. Aide au développement durable des territoires: approche muli-agents pour un modèle enjeux/acteurs. In *Modèles et systems multi-agents pour la gestion de l'environment et des territoires*, N. Ferrand, Ed. Cemagraf, Paris.

Georgeff, M. P., Pell, B., Pollack, M., Tambe, M., and Wooldridge, M. 1999. The belief-desire-intention model of agency. *Lecture 1555*, 1–10.

Gilbert, N. 2006. When does social simulation need cognitive models? In *Cognition and Multi-Agent Interaction*, R. Sun, Ed. Cambridge University Press, Chapter 19, 428–432.

Gimblett, H., Ed. 2002. *Integrating geographic information systems and agent-based modeling techniques for simulating social and ecological processes*. Oxford University Press, USA.

Gobelbecker, M. and Dornhege, C. 2009. Realistic cities in robocup rescue - an open street map to rescue converter.

Gonçalves, A., Rodrigues, A., and Correia, L. 2004. Multi-agent simulation within geographic information systems. In *Proceedings of the 5th International Workshop on Agent-Based Simulation (ABS-2004)*. 107–112.

Gonzalez, R. A. 2009a. Analysis and design of a multi-agent system for simulating a crisis response organization. In *Proceedings of the EOMAS Workshop held in conjunction with CAiSE'09 Conference*, J. Barjis, J.Kinghorn, S. Ramaswamy, E. Dubois, and P. Johannesson, Eds. Amsterdam, The Netherlands.

Gonzalez, R. A. 2009b. Crisis response simulation combining discrete-event and agent-based modeling. In *Proceeding of the 6th International ISCRAM Conference*, J. L. . S. Jul, Ed. Gothenburg, Sweden.

Gonzalez, R. A. 2010. A Framework for ICT-Supported coordination in Crisis Response. Ph.D. thesis, Delft University of Technology.

Grimm, V. and Railsback, S. F. 2005. *Individual-based Modeling and Ecology*. Princeton University Press.

Grimm, V., Revilla, E., Berger, U., Jeltsch, F., Mooij, W. M., Railsback, S. F., Thulke, H.-H., Weiner, J., Wiegand, T., and DeAngelis, D. L. 2005. Pattern-Oriented Modeling of Agent-Based Complex Systems: Lessons from Ecology. *Science 310*, 987–991.

Gropp, W., Lusk, E., and Skjellum, A. 2000. *Using MPI: Portable Parallel Programming with the Message-Passing Interface*, 2nd ed. MIT Press.

Gross, D. and Strand, R. 2000. Can agent-based models assist decisions on large-scale practical problems? a philosophical analysis. *Complexity 5,* 6, 26–33.

Gulyas, L., Szemes, G., Kampis, G., and de Back, W. 2009. A Modeler-Friendly API for ABM Partitioning. In *Proceedings of the ASME 2009 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2009*.

Guo, D., Ren, B., and Wang, C. 2008. Integrated agent-based modeling with GIS for large scale emergency simulation. In *Proceedings of the 3rd International Symposium on Advances in Computation and Intelligence*. Springer, E1–E2.

Guttman, A. 1984. R-trees: a dynamic index structure for spatial searching. In *Proceedings of the 1984 ACM SIGMOD international conference on Management of data*.

Haddow, G. D., Bullock, J. A., and Coppola, D. P. 2010. *Introduction to Emergency Management*. Butterworth-Heinemann.

Hager, G. and Wellein, G. 2011. *Introduction to High Performance Computing for Scientists and Engineers*. CRC Press.

Hallet, H. 2011. Coroner's Inquests into the London Bombings of 7 July 2005. http://7julyinquests.independent.gov.uk/docs/orders/rule43-report.pdf.

Hare, M. and Deadman, P. 2004. Further towards a taxonomy of agent-based simulation models in environmental management. *Mathematics and Computers in Simulation 64,* 1, 25–40.

Hart, P. E. and Raphael, N. J. N. B. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics 4,* 2, 100–107.

Helbing, D. et al. 2010. The FuturIcT Knowledge Accelerator: Unleashing the Power of Information for a Sustainable Future. http://arxiv.org/ftp/arxiv/papers/1004/1004.4969.pdf.

Howe, T., Collier, N., North, M., M.T.Parker, and Vos, J. 2006. Containing Agents: Contexts, Projections and Agents. In *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects*, C. M. D. Sallach and M. North, Eds.

Hunt, R., Brown, L., Cabinum, E., Whitley, T., Prasad, N., Owens, C., et al. 1995. Is ambulance transport time with lights and siren faster than that without? *Annals of Emergency Medicine 25,* 4, 507–511.

Jacobs, P., Lang, N., and Verbraeck, A. 2002. D-SOL: a distributed Java based discrete event simulation architecture. In *Proceedings of the 34th conference on Winter simulation: exploring new frontiers*. Winter Simulation Conference, 793–800.

JADE. 2010. Java Agent DEvelopment Framework. http://jade.tilab.com/.

Jain, S. and McLean, C. 2003. A framework for modeling and simulation for emergency response. In *Proceedings of the 2003 Winter Simulation Conference*, S. Chick, P. J. Snchez, D. Ferrin, and D. J. Morrice, Eds. 1068–1078.

Jess. 2008. The rule engine for the java platform. http://www.jessrules.com/.

JOSM. 2010. Java OpenStreetMap Editor. http://josm.openstreetmap.de/.

JSI. 2010. Java spatial index. http://jsi.sourceforge.net/.

JTS. 2006. JTS Topology Suite. http://www.vividsolutions.com/JTS/JTSHome.htm.

Kennedy, R., Lane, K., Fuentes, A., Hollocher, H., and Madey, G. 2009. Spatially aware agents: an effective and efficient use of GIS data within an agent-based model. In *Proceedings of the 2009 Spring Simulation Multiconference*. Society for Computer Simulation International, 1–8.

Kermack, W. O. and McKendrick, A. G. 1927. A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society of London A 115*, 700–721.

Khalil, K., Abdel-Aziz, M., Nazmy, T., and Salem, A. 2009. Bridging the Gap between Crisis Response Operations and Systems. *Arxiv preprint arXiv:0908.4290*.

Khalil, K. M., Abdel-Aziz, M., Nazmy, T. T., and Salem, A.-B. 2010. An agent-based modeling for pandemic influenza in egypt. In *Proceedings of the 7th International Conference on Informatics and Systems (INFOS)*.

Kim, T., Hwang, W., Zhang, A., Sen, S., and Ramanathan, M. 2008. Multi-agent model analysis of the containment strategy for avian influenza (ai) in south korea. In *Bioinformatics and Biomedicine, 2008. BIBM '08. IEEE International Conference on*. 335 –338.

Kirk, D. B. and Hwu, W. W. 2010. *Programming Massively Parallel Processors: A Hands-on Approach*. Morgan Kaufmann.

Kitano, H., Tadokoro, S., Noda, I., Matsubara, H., Takahashi, T., Shinjou, A., and Shimada, S. 1999. RoboCup Rescue: search and rescue in large-scale disasters as a domain for autonomous agents research. In *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics, 1999.* Vol. 6.

Klein, G. 2007a. Flexecution as a paradigm for replanning. *IEEE Intelligent Systems 22,* 5, 79–83.

Klein, G. 2007b. Flexecution, part 2: Understanding and supporting flexible execution. *IEEE Intelligent Systems 22,* 6, 108–112.

Klein, G. 2008. Naturalistic decision making. *Human Factors 50,* 3 (June), 456–460.

Klein, G. A., Calderwood, R., and Clinton-Cirocco, A. 1986. Rapid decision making on the fire ground. In *Human Factors and Ergonomics Society Annual Meeting Proceedings*. Vol. 30. Human Factors and Ergonomics Society, 576–580.

Kleiner, A., Behrens, N., and Kenn, H. 2006. Wearable computing meets multiagent systems: A real-world interface for the robocuprescue simulation platform. In *Proceedings of the Agent Technology for Disaster Management (ATDM) Workshop, 2006*. Hakodate, Japan.

Kleiner, A., Brenner, M., Brauer, T., Dornhege, C., Gobelbecker, M., Luber, M., Prediger, J., Stuckler, J., and Nebel, B. 2006. Successful search and rescue in simulated disaster areas. *Lecture Notes in Artificial Intelligence 4020*, 323–334.

Korf, R. E. 1990. Real-time heuristic search. *Artificial Intelligence 42*, 189–211.

Koto, T. and Takeuchi, I. 2003. A distributed disaster simulation system that integrates subsimulators. In *First International Workshop on Synthetic Simulation and Robotics to Mitigate Earthquake Disaster*. Citeseer.

Koza, J. R. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press.

Lacher, M. and Bausher, J. 1997. Lights and siren in pediatric 911 ambulance transports: Are they being misused? *Annals of emergency medicine 29,* 2, 223–227.

LAS. 2011. London ambulance service latest response times. http://www.londonambulance.nhs.uk/about_us/how_we_are_doing/meeting_our_targets/latest_response_times.aspx.

Law, A. M. and Kelton, W. D. 1982. *Simulation Modeling and Analysis*, 1st ed. McGraw-Hill.

Lee, S. W., Davidson, R. A., and Lembo, A. J. 2008. Gis-based algorithms in support of post-earthquake fire spread modeling. www.urisa.org/files/Lee_PostEarthquake_FireSpreadModeling.doc.

Lerner, E. and Moscati, R. 2001. The golden hour: scientific fact or medical 'urban' legend? *Academic Emergency Medicine 8,* 7, 758–760.

Liebert, K., Earnest, D., and Tolk, A. 2008. Using GIS vector data to build virtual environments for agent based models. In *Proceedings of the 2008 Spring simulation multiconference*. The Society for Computer Simulation, International San Diego, CA, USA, 45–52.

Lin, M. C. and Manocha, D. 2010. Simulation technologies for evacuation planning and disaster response. Tech. rep., Institute for Homeland Security Solutions.

London Resilience Team. 2010. London Regional Command and Control Protocol. http://www.londonprepared.gov.uk/downloads/LondonC-C-Protocol-V4.pdf. Version 4.

Macal, C. M. and North, M. J. 2010. Tutorial on agent-based modelling and simulation. *Journal of Simulation 4*, 151–162.

Maguire, B., Hunting, K., Smith, G., and Levick, N. 2002. Occupational fatalities in emergency medical services: a hidden crisis. *Ann Emerg Med 40,* 6, 625–632.

Manolopoulos, Y., Nanopoulos, A., Papadopoulos, A. N., and Theodoridis, Y. 2003. R-trees have grown everywhere. http://www.rtreeportal.org/.

Massaguer, D., Balasubramanian, V., Mehrotra, S., and Venkatasubramanian, N. 2006. Multi-agent simulation of disaster response. In *Proceedings of the First International Workshop on Agent Technology for Disaster Management, ATDM*. Citeseer.

Massaioli, F., Castiglione, F., and Bernaschi, M. 2005. OpenMP parallelization of agent-based models. *Parallel Computing 31,* 10-12, 1066–1081.

Mendonça, D. 2007. Decision support for improvisation in response to extreme events: Learning from the response to the 2001 World Trade Center attack. *Decision Support Systems 43,* 3, 952–967.

Mendonça, D. and Fiedrich, F. 2006. Training for improvisation in emergency management: Opportunities and limits for information technology. *International Journal of Emergency Management 3,* 4, 348–363.

Motowidlo, S. J., Dunnette, M. D., and Carter, G. W. 1990. An alternative selection procedure: the low-fidelity simulation. *Journal of Applied Psychology 75*, 640–647.

Mukherjee, T. and Gupta, S. K. S. 2009. CRET: A Crisis Response Evaluation Tool to improve Crisis Preparedness. In *Proceedings of IEEE International Conference on Technologies for Homeland Security*. Citeseer.

Müller, J. P. 1999. The right agent (architecture) to do the right thing. *Lecture Notes in Computer Science 1555*, 211–225.

Mysore, V., Gill, O., Daruwala, R., Antoniotti, M., Mishra, B., and Saraswat, V. 2005. Multi-agent modeling and analysis of the brazilian food poisoning scenario. In *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms*, C. Macal, M. North, and D. Sallach, Eds.

Mysore, V., Narzisi, G., and Mishra, B. 2006. Agent Modeling of a Sarin Attack in Manhattan. In *Proceedings of the First International Workshop on Agent Technology for Disaster Management*. Hokkaido, Japan.

Narzisi, G., Mincer, J., Smith, S., and Mishra, B. 2007. Resilience in the Face of Disaster: Accounting for Varying Disaster Magnitudes, Resource Topologies, and (Sub) Population Distributions in the Plan C Emergency Planning Tool. *Lecture Notes in Computer Science 4659*, 433.

Narzisi, G., Mysore, V., and Mishra, B. 2006. Multi-objective evolutionary optimization of agent based models: an application to emergency response planning. In Proceedings of the Second IASTED International Conference on Computational Intelligence. *Computational Intelligence-CI*, 224–230.

Narzisi, G., Mysore, V., Nelson, L., Rekow, D., Triola, M., Halcomb, L., Portelli, I., and Mishra, B. 2006. Complexities, Catastrophes and Cities: Unraveling Emergency Dynamics. In *Proceedings of Sixth International Conference on Complex Systems (ICCS 2006)*, A. Minai, D. Braha, and Y. Bar-Yam, Eds.

Newgard, C. D., Schmicker, R. H., Hedges, J. R., Trickett, J. P., Davis, D. P., Bulger, E. M., Aufderheide, T. P., Minei, J. P., Hata, J. S., Gubler, K. D., Brown, T. B., Yelle, J.-D., Bardarson, B., and Nichol, G. 2010. Emergency medical services intervals and survival

in trauma: Assessment of the golden hour in a north american prospective cohort. *Annals of Emergency Medicine 55,* 3, 235–246.

Nikolai, C. and Madey, G. 2007. Anatomy of a toolkit: A comprehensive compendium of various agent-based modeling toolkits on the market today. In *Proceedings of the Agent 2007 Conference on Complex Interaction and Social Emergence*, M. North, C. Macal, and D. Sallach, Eds. 87–98.

Nikolai, C. and Madey, G. 2009a. Searchable Taxonomies of Agent Based Modeling Toolkits. In *Proceedings of the 2009 Spring Simulation Multiconference.*

Nikolai, C. and Madey, G. 2009b. Tools of the Trade: A Survey of Various Agent Based Modeling Platforms. *Journal of Artificial Societies and Social Simulation 12,* 2, 2.

North, M. J. and Macal, C. M. 2007. *Managing Business Complexity: Discovering Strategic Solutions with Agent-Based Modeling and Simulation.* Oxford University Press: New York, NY.

North, M. J., Tatara, E., Collier, N. T., and Ozik, J. 2007. Visual agent-based model development with repast simphony. In *Proceedings of the Agent 2007 conference on Complex Interaction and Social Emergence.*

Nussle, T. A., Kleiner, A., and Brenner, M. 2005. Approaching Urban Disaster Reality: The ResQ Firesimulator. *Lecture Notes in Computer Science 3276*, 474–482.

OpenStreetMap. 2010. The Free Wiki World Map. http://www.openstreetmap.org/.

Ören, T. and Longo, F. 2008. Emergence, anticipation and multisimulation: Bases for conflict simulation. In *Proceedings of the 20th European Modeling & Simulation Symposium (EMSS).* Campora San Giovanni, Italy, 546–555.

Orion. 2010. Exercise Orion. http://www.orion2010.co.uk/.

Osterwalder, J. 2002. Can the 'golden hour of shock' safely be extended in blunt polytrauma patients? Prospective cohort study at a level I hospital in eastern Switzerland. *Prehospital and disaster medicine 17,* 2, 75–80.

O´Sullivan, D. 2008. Geographical information science: agent-based models. *Progress in Human Geography 32,* 4, 541–550.

Oğuz, O., Akaydin, A., Yilmaz, T., and Güdükbay, U. 2010. Emergency crowd simulation for outdoor environments. *Computers & Graphics 34,* 2, 136–144.

Parker, J. 2007. A flexible, large-scale, distributed agent based epidemic model. In *Simulation Conference, 2007 Winter.* IEEE, 1543–1547.

Pawling, A., Schoenharl, T., Yan, P., and Madey, G. 2008. WIPER: An Emergency Response System. In *Proceedings of the 5th International ISCRAM Conference*, F. Fiedrich and B. V. de Walle, Eds.

Petri, R., Dyer, A., and Lumpkin, J. 1995. The effect of prehospital transport time on the mortality from traumatic injury. *Prehospital and disaster medicine: the official journal of the National Association of EMS Physicians and the World Association for Emergency and Disaster Medicine in association with the Acute Care Foundation 10,* 1, 24.

Petrie, C. 2007. No Science without Semantics. *IEEE Internet Computing 11*, 88, 86–87.

Pokahr, A., Braubach, L., and Lamersdorf, W. 2005. Jadex: A BDI Reasoning Engine. In *Multi-Agent Programming*, G. Weiss, K. M. Carley, Y. Demazeau, E. Durfee, L. Gasser, N. Gilbert, M. Huhns, N. Jennings, V. Lesser, K. Sycara, M. Wooldridge, R. Bordini, M. Dastani, J. Dix, and A. Fallah Seghrouchni, Eds. Multiagent Systems, Artificial Societies, and Simulated Organizations, vol. 15. Springer US, 149–174.

Pons, P. and Markovchick, V. 2002. Eight minutes or less: does the ambulance response time guideline impact trauma patient outcome? *Journal of Emergency Medicine 23,* 1, 43–48.

Pooyandeh, M., Mesgari, S., and Alimohammadi, A. 2007. Integration of Agent-based Modeliing and GIS for Urban Simulation. In *Proceedings of Geomatics 1386.*

Railsback, S. F., Lytinen, S. L., and Jackson, S. K. 2006. Agent-based Simulation Platforms: Review and Development Recommendations. *Simulation 82,* 9 (September), 609–623.

Rand, W., Brown, D., Riolo, R., and Robinson, D. 2005. Toward a Graphical ABM Toolkit with GIS Integration. In *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms*, C. Macal, M. North, and D. Sallach, Eds. 27–42.

Ren, C., Yang, C., and Jin, S. 2009. Agent-Based Modeling and Simulation on Emergency Evacuation. In *Complex Sciences*. Springer Berlin Heidelberg, 1451–1461.

Repast. 2010. The Repast Suite. http://repast.sourceforge.net/.

Reuters. 2011. Hawaii orders evacuations in pacific tsunami threat. http://www.reuters.com/article/2011/03/11/us-japan-quake-tsunami-hawaii-idUSTRE72A1OW20110311.

Roberts, D. 2010. Distributed agent based modeling. http://www.linuxjournal.com/content/distributed-agent-based-modeling.

Robocup Rescue Simulation Project. 2010. http://www.sf.net/projects/roborescue.

Ruas, T., das Gracas B. Marietto, M., dos S. Franca, R., and de M. Batista, A. F. 2009. A model for fire spreading by multi-agent systems: A robocup rescue simulation and swarm platform approach. In *Second International Conference on the Applications of Digital Information and Web Technologies, 2009. ICADIWT '09.* London, 380–385.

Rubinstein, A. 1998. *Modeling bounded rationality.* MIT Press.

Runka, A. 2010. Genetic Programming for the RoboCup Rescue Simulation System. M.S. thesis, Faculty of Computer Science, Brock University, St. Catharines, Ontario.

Russell, S. and Norvig, P. 2009. *Artificial Intelligence: A Modern Approach.* Prentice Hall.

Samuelson, D. A., Parker, M., Zimmerman, A., Miller, L., Guerin, S., Thorp, J., and Densmore, O. 2008. Agent-bsed simualations of mass egress after improvised explosive device attacks. In *Proceedings of the 5th International ISCRAM Conference*, F. Fiedrich and B. V. de Walle, Eds.

Saoud, N. B.-B., Darcy, S., Dugdale, J., Pavard, B., and Ahmed, M. B. 2003. Simulation multi-agents de situation de secours d'urgence (multi-agent simulation of emergency situations). In *Proceedings of the 10th Journees Francophones d' Informatique Medicale.* Tunis.

Saoud, N. B.-B., Dugdale, J., Pavard, B., Ahmed, M. B., Mena, T. B., and Touai, N. B. 2004. Towards planning for emergency activities in large-scale accidents. In *Proceedings of ISCRAM.* Brussels.

Saoud, N. B.-B., Mena, T. B., Dugdale, J., Pavard, B., and Ahmed, M. B. 2006. Assessing large scale emergency rescue plans: An agent based approach. *The International Journal of Intelligent Control and Systems 11,* 4, 260–271.

Saoud, N. B.-B., Pavard, B., and Dugdale, J. 2005. An agent-based testbed for simulating large scale accident rescue heuristics. *Journal of Computer Science Special Issue*, 21–26.

Sarika, R. 2010. Database driven robocup rescue server. Ph.D. thesis, International Institute of Information Technology, Hyderabad, India.

Sato, K. and Takahashi, T. 2011. A study of map data influence on disaster and rescue simulation's results. In *Advances in Practical Multi-Agent Systems*, Q. Bai and N. Fukuta, Eds. Studies in Computational Intelligence, vol. 325. Springer Berlin / Heidelberg, 389–402.

Sawaragi, Y., Nakayama, H., and Tanino, T. 1985. *Theory of Multiobjective Optimization.* Academic Press Inc.

Schoenharl, T., Bravo, R., and Madey, G. 2006. WIPER: Leveraging the cell phone network for emergency reponse. *International Journal of Intelligent Control and systems 11,* 4, 209–216.

Schoenharl, T., Zhai, Z., McCune, R., Pawling, A., and Madey, G. 2009. Design and implementation of an agent-based simulation for emergency response and crisis management. In *Proceedings of the 2009 Spring Simulation Multiconference.*

Schoenharl, T. W. 2007. Creating, updating and validating simulations in a dynamic data-driven application system. Ph.D. thesis, University of Notre Dame.

Schule, M., Herrler, R., and Klugl, F. 2004. Coupling GIS and Multi-agent Simulation Towards Infrastructure for Realistic Simulation. *Lecture Notes in Computer Science 3187*, 228–242.

Sengupta, R. and Sieber, R. 2007. Geospatial Agents, Agents Everywhere... *Transactions in GIS 11,* 4, 483–506.

Sethia, P. and Karlapalem, K. 2010. RoboCup Rescue Simulator on Graphics Processing Units. http://roborescue.sourceforge.net/2010/tdps/infra/gpu.pdf.

Shahbazi, H., Abdolmaleki, A., Salehi, S., Shahsavari, M., and Movahedi, M. 2010. Brave Circles Team Description Paper. http://roborescue.sourceforge.net/2010/teams.html.

Siddhartha, H., Sarika, R., and Karlapalem, K. 2009a. Retrospective Analysis of RoboCup Rescue Simulation Agent Teams. In *Proceedings of Autonomous Agents and Multiagent Systems (AAMAS2009)*, S. Decker, Sichman and Castelfranchi, Eds. Budapest, Hungary.

Siddhartha, H., Sarika, R., and Karlapalem, K. 2009b. Score Vector: A New Evaluation Scheme for RoboCup Rescue Simuation Competition 2009.

Simon, H. 1957. A behavioral model of rational choice. In *Models of Man, Social and Rational: Mathematical Essays on Rational Human Behavior in a Social Setting*. New York: Wiley.

Skinner, C. and Ramchurn, S. 2010. The RoboCup Rescue Simulation Platform. In *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lesperance, Luck, and Sen, Eds. 1647–1648.

Skvortsov, A., Connell, R., Dawson, P., and Gailis, R. 2007. Epidemic modelling: Validation of agentbased simulation by using simple mathematical models. In *MODSIM 2007 International Congress on Modelling and Simulation*. Citeseer.

Smith, L., Beckman, R., and Baggerly, K. 1995. TRANSIMS: Transportation analysis and simulation system. Tech. rep., Los Alamos National Lab., NM (United States).

Stentz, A. 1995. The Focussed D* Algorithm for Real-Time Replanning. In *International Joint Conference on Artificial Intelligence*. Vol. 14. Citeseer, 1652–1659.

Stirling, W. C. 2003. *Satisficing Games and Decision Making*. Cambridge University Press.

Straylight. 2010. The Emergency Simulation Program (ESP). http://www.straylightmm.com/.

Sun, R. 2006. Prolegomena to integrating cognitive modeling and social simulation. In *Cognition and Multi-Agent Interaction*, R. Sun, Ed. Cambridge University Press, Chapter 1, 3–28.

Sutter, H. 2005. The free lunch is over : A fundamental turn toward concurrency in software. *Dr. Dobb's Journal 30,* 3 (March), 202–210.

Sutter, H. and Larus, J. 2005. Software and the concurrency revolution. *Queue 3,* 7, 54–62.

Swarm. 2011. Swarm agent platform. http://www.swarm.org/index.php/Main_Page.

Tadokoro, S. 2006. Robocup-rescue simulation project overview. http://www.rescuesystem.org/robocuprescue/simoverview.html.

Tadokoro, S., Kitano, H., Takahashi, T., Noda, I., Matsubara, H., Shinjoh, A., Koto, T., Takeuchi, I., Takahashi, H., Matsuno, F., Hatayama, M., Nobe, J., and Shimada, S. 2000. The robocup-rescue project: A robotic approach to the disaster mitigation problem. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA'00.* Vol. 4. IEEE, San Francisco, CA , USA, 4089–4094.

Takahashi, H., Tanigawa, M., and Takahashi, T. 2005. Tool kits for using Open Source GIS data as RoboCup Rescue GIS maps. In *Proceedings of RoboCup 2005*.

Takahashi, T. 2007. Agent-based disaster simulation evaluation and its probability model interpretation. In *Proceedings of ISCRAM2007*. 369–376.

Takahashi, T. 2009. RoboCup Rescue - Agent-based Disaster Simulation System: Challenges and Lessons Learned. In *Multi-Agent Systems: Simulation and Applications*, A. M. Uhrmacher and D. Weyns, Eds. CRC Press.

Takahashi, T., Takeuchi, I., Koto, T., Tadokoro, S., and Noda, I. 2001. Robocup Rescue Disaster Simulator Architecture. *Lecture Notes in Computer Science 2019*, 379–384.

Takeuchi, I. 2005. A massively multi-agent simulation system for disaster mitigation. In *Massively multi-agent systems I: first international workshop, MMAS 2004, Kyoto, Japan, December 10-11, 2004: revised selected and invited papers*. Springer, 269.

Tanigawa, M., Takahashi, T., Koto, T., Takeuchi, I., and Noda, I. 2005. Urban flood simulation as a component of integrated earthquake disaster simulation. In *Proceedings of the 2005 IEEE International Safety, Security and Rescue Robotics Workshop*. IEEE, 248–252.

Terracotta. 2010. http://www.terracotta.org/solutions/grid.

Turner, J., OKeefe, C., Dixon, S., Warren, K., and Nicholl, J. 2006. The costs and benefits of changing ambulance response time performance standards. Tech. rep., Medical Care Research Unit School of Health and Related Research, University of Sheffield.

Turton, I. 2008. GeoTools. In *Open Source Approaches in Spatial Data Handling*, G. Hall and M. Leahy, Eds. Advances in Geographic Information Science, vol. 2. Springer Berlin Heidelberg, Chapter 8, 153–169.

Uchmánski, J. and Grimm, V. 1996. Individual-based modelling in ecology: What makes the difference? *Trends in Ecology and Evolution 11*, 437–441.

U.K. Cabinet Office. 2010a. National Risk Register of Civil Emergencies. http://www.cabinetoffice.gov.uk/media/348986/nationalriskregister-2010.pdf.

U.K. Cabinet Office. 2010b. Responding to Emergencies, The UK Central Government Response: Concept of Operations. http://www.cabinetoffice.gov.uk/media/349120/conops-2010.pdf.

U.K. Cabinet Office. 2011. Emergency Exercises. http://www.cabinetoffice.gov.uk/content/emergency-exercises.

USA Today. 2011. La. floodgate opened; residents brace for flooding. http://www.usatoday.com/weather/floods/2011-05-13-louisiana-spillway_n.htm.

Valle, S. D., Kubicek, D., Mniszewski, S., Riese, J., Romero, P., Smith, J., Stroud, P., and Sydoriak., S. 2006. EpiSimS Los Angeles Case Study. Tech. Rep. LAUR-06-0666, Los Alamos National Laboratory. January.

Vapnik, V. 2006. *Estimation of Dependences Based on Empirical Data*. Springer.

Wang, J., Xiong, J., Yang, K., Peng, S., and Xu, Q. 2010. Use of GIS and agent-based modeling to simulate the spread of influenza. In *Proceedings of the 18th International Conference on Geoinformatics*. 1 –6.

Warshall, S. 1962. A theorem on Boolean matrices. *Journal of the ACM 9,* 1 (January), 11–12.

Wolf, E. 2003. Using agent-based distillations to explore logistics support to urban, humannitarian assistance/disaster relief operations. Ph.D. thesis, Naval Postgraduate School.

Wooldridge, M. 2009. *An Introduction to Multi-Agent Systems*, 2nd ed. John Wiley & Sons.

Wooldridge, M. and Jennings, N. R. 1995. Intelligent agents: theory and practice. *The Knowledge Engineering Review 10,* 2, 115–152.

Xiang, X., Kennedy, R., Madey, G., and Cabaniss, S. 2005. Verification and validation of agent-based scientific simulation models. In *Proceedings of the 2005 Agent-Directed Simulation Symposium*, L. Yilmaz, Ed. 47–55.

Yahja, A. and Carley, K. 2006. WIZER: Automated model improvement in multi-agent social-network systems. In *Coordination of Large-Scale Multiagent Systems*, P. Scerri, R. Vincent, and R. Mailler, Eds. Springer, 255–270.

Yoo, M.-J. and Glardon, R. 2009. Combining JADE and Repast for the Complex Simulation of Enterprise Value-Adding Networks. *Lecture Notes in Computer Science 5386*, 243–256.

Zhao, K., Yen, J., Maitland, C., Tapia, A., and Tchouakeu, L. 2009. A formal model for emerging coalitions under network influence in humanitarian relief coordination. In *Proceedings of the 2009 Spring Simulation Multiconference*. International Society for Computer Simulation.

Zhong, W. and Kim, Y. 2011. Searching for sweet spots of communication during an emergency. In *The 11th Biennial and 20th Anniversary Public Management Research Conference*.