Submitted to *Future Internet*. Pages 1 - 19.

OPEN ACCESS

*Article*

# Managing Emergencies Optimally using a Random Neural Network Based Algorithm

**Qing Han**

Intelligent Systems & Networks Group, EEE, Imperial College, London SW7 2BT, UK

* Author to whom correspondence should be addressed; {qing.han12}@imperial.ac.uk

*Version August 29, 2013 submitted to* FutureInternet. *Typeset by LaTeX using class file mdpi.cls*

**Abstract:** Emergency rescues require that first responders provide support to evacuate injured and other civilians who are obstructed by the hazards. In this case, the emergency personnel can take actions strategically in order to rescue people maximally, efficiently, and quickly. The paper studies the effectiveness of a Random Neural Network (RNN) based task assignment algorithm involving optimally matching emergency personnel and injured civilians, so that the emergency personnel can aid trapped people to move towards evacuation exits in real-time. The system is evaluated in various emergency scenarios using the Distributed Building Evacuation Simulator (DBES) multi-agent platform. The simulations indicate that the RNN based task assignment algorithm provides a near-optimal solution to resource allocation problems, which avoids the resource wastage and improves the efficiency of emergency rescue process.

**Keywords:** Rescuers, Random Neural Network (RNN), Task Assignment

## 1. Introduction

In real emergency environments, unprotected evacuees may be seriously injured and even immobilized due to the hazards, or may be trapped in the disasters and find there is no way out [1]. As a result, they need emergency personnel [2] to provide assistance and take them out of the danger. Rescuers can move to the target locations quickly and protect themselves and evacuees from the danger by wearing special clothes and using protective devices, so that they can make great contribution to saving people out of the hazards [1]. However, the rescuers may run

into the danger and fail in saving a victim since they move into the hazardous sites, and they may not know which victim has been allocated a rescuer or which rescuer has been assigned a victim if there is no interaction between each assignment. These uncertain factors lead to a waste of resources (e.g. different rescuers are allocated to the same victim) and the inefficiency of resource allocations (e.g. a rescuer is unable to save a victim though the rescuer has been dispatched to it, and different victims are assigned to the same rescuer). Moreover, during an ongoing hazard, the decision-making of how to allocate rescuers to victims requires a fast computational algorithm to provide advice in real-time. Therefore, a fast and efficient assignment approach which considers the possibility of unsuccessful rescue and the influence of other assignments should be addressed.

The paper investigates the problem of how to dispatch rescuers to save people out of the danger efficiently when a limited set of rescuers needs to be allocated simultaneously to a set of injured civilians [3]. This is a kind of task assignment problem which aims to optimally match the elements of two or more sets and reduce the overall cost of associated matchings [4]. Here, only two sets will be considered, which we refer to as "tasks (i.e. injured civilians)" and "resources (i.e. rescuers)". Previous work [3] proposes that the random neural network can be applied to resource allocation problems arising in emergency management to provide a decision quickly and exactly. In the RNN, every possible assignment is considered to be a neuron, and all neurons in the network are fully connected, that is, each executed assignment can have an effect on other allocations through the links between them. Also, there is a cost objective function which can be formulated considering any uncertain parameters (i.e. RNN parameters) defined as needed, and an optimal assignment will minimize the expected cost. In this case, we can pick out an assignment which can maximally reduce the overall expected cost by updating RNN parameters iteratively until convergence, and then this assignment will be executed. The detail of the RNN algorithm will be introduced in Section 3.

The DBES multi-agent environment can capture the position and action of people in the area by using RFID devices and audiovisual sensors [5], thus, if the evacuees are at stake in the building, the sensors will send a message to the rescuers and ask for help. The message received by the rescuers can include the detailed information of the injured civilians, such as their ID, position, and nature of injury. By applying the RNN based algorithm, the assignment with the highest active probability will be executed. The duty of rescuers is then to search for the injured civilians relying on the information received, and lead or take them to the best evacuation exit or assembly point. The rescuers will wear protective clothes to be less susceptible to injuring themselves, and they will carry protective devices to assist evacuees.

The remainder of the paper is organized as follows. Section 2 summarizes the related work on emergency management and the RNN based techniques. Section 3 describes the design details of the RNN based task assignment algorithm. Then the simulation results are presented in Section 4, in which we show that the RNN based algorithm can provide a near-optimal solution to resource allocation problems arising in emergency management. Finally, in Section 5 we conclude the results, and discuss the remaining problems of the algorithm and possible directions for future work.

## 2. Literature Review

In the paper, we investigate the resource allocation problem for emergency management, which covers the areas of evacuation navigation, distributed simulator, analytical performance model, and the RNN based algorithms. We discuss the related work in each of these areas below.

A distributed navigation algorithm proposed in [6] is applied in mobile networks to guide people to a exit safely during an emergency with a 2D layout, and the work is extended to 3D indoor environment in [7]. [8] presents a distributed algorithm to find safe paths for evacuees which takes into account predictions of the movement of the hazards, but there are some shortcomings addressed in [9], where the authors extend the concept safety introduced in [8]. The distributed building evacuation simulator used in our simulations is firstly proposed in [10] and particularly in [11]. It operates in a distributed fashion to reduce the simulation time required for large-scale systems and facilitate the modelling of the agents that need to be simulated. Previous work [12] discusses the use of analytical performance models, in which they consider the complex programming problem and enhance previous methodologies, and in [13] a more recent analytical model is introduced, which can provide fast estimates for path discovery, for the sensitivity of outcomes to the presence of hazards, and for the location of congestion in novel circumstances. Moreover, based on the work [10–13], [14] illustrates that routing emergency evacuees with Cognitive Packet Network (CPN) [15–17] can alleviate slow convergence time for large and fast-changing networks, and [18] proposes the use of Opportunistic Communications (oppcomms) which can provide emergency support when a stable communication infrastructure is unavailable.

The above work attempts to direct the evacuees to get out the danger quickly and avoid injury, however, people might be seriously injured and obstructed by the hazards in real emergency environments. Thus, a system which can efficiently search for the injured civilians and coordinate rescue is necessary for emergency management. In [19] and [20], the authors address a robot-sensor network to identify the locations of possible victims for providing rescue to those trapped people. A distance-sensitive service discovery algorithm is illustrated in [21], which combines the concepts of Voronoi polygons [22] and the quorum-based approaches [23], and guarantees closest tasks (e.g. victims) being executed with very high probability. By considering the energy issue, [24] explores two algorithms to dispatch mobile resources (e.g. emergency units) to visit task locations in an energy-balanced way. Also, [25] takes into account the situation where people may be trapped with no escape path due to the congestion, and designs a scheme for allocating resources to eliminate key dangerous areas in order to ease congestion and increase the number of rescued people. Furthermore, the authors in [26] consider that a resource may fail in executing a task in dynamic hazard environments, and address a task assignment algorithm with a probabilistic successful execution outcome. The algorithm is on the basis of the RNN techniques so that it is able to solve the resource allocation problems accurately and approximate in real-time. The random neural network is powerful in obtaining desired input-output mappings through learning and has been widely used in solving real world problems. The theory and some applications of the RNN is introduced below.

The excitatory and inhibitory signals concept of the RNN is first introduced in [27] where each neuron has an active potential which will be affected by the input signals from other neurons and the outside world. The neuron will be excited if its active potential is a positive value, and if a neuron is excited, it will send excitatory or inhibitory signals at an excited rate to other neurons or to the outside world. The RNN has product form in the Markovian case and leads to a concise representation for its steady-state behaviour. However, the signal flow equations of the model is nonlinear, so that the unique solutions are only obtained in the case of feedforward networks. Thus, [28] establishes the uniqueness of solutions in the networks with feedback and shows that the signal flow equations have a unique solution whenever the solution exists. In [29], the authors illustrates that the RNN with feedforward structures can provide a good approximation to input-output matching problems, while [30] addresses that the feedforward RNN with more constraints also can uniformly approximate a continuous function. In engineering applications, the need to process different classes of information simultaneously is required, hence [31] proposes a extension to the RNN model which includes multiple classes of signals and preserves the existence and uniqueness of the product form solution of the network. A learning algorithm used in multiple class random neural networks is introduced in [32], which is capable of learning sets of given input-output examples and then making a prediction on input-output mappings under new environments. Another extension of the RNN is discussed in [33], where the network incorporates the usual excitatory and inhibitory information but also creates the probability that a synchronous interaction between two neurons affects some third neuron. The network also has product form and preserves the unique solution property.

One kind of application of the RNN is the image processing and video compression. [34] describes a scheme for real-time adaptive compression and decompression of video based on the RNN, and [35] presents a geometric recurrent neural network to solve magnetic resonance imaging (MRI) brain morphometry problems effectively and efficiently. Furthermore, the RNN can discover a new route in a self-aware manner. In [17], a self-aware network (SAN) based on the RNN techniques is introduced, where the nodes are able to discover the network state autonomously without relying on an overall routing table and compute the next hop as needed and on demand. Based on the work in [17], [36] presents a measurement-based admission control algorithm to make sure packets delivery occur under Quality of Service (QoS) constraints, and [37] proposes an autonomic approach to defend against Denial of Service (DoS) attacks. Additionally, the authors in [38] apply a genetic algorithm (GA) on the source nodes, which can modify and select the paths discovered by the SAN on the basis of their own needs. The RNN has also been suggested as a tool to make decisions quickly and accurately for optimization problems. The paper [39] develops a dynamic random neural network (DRNN) approach to solve the traveling salesman problem optimally, and the results verify that the DRNN can provide substantially better performances compared with other neural networks. Also, the RNN can be equipped to route multicast communications in an efficient manner by constructing a minimum Steiner tree, which is presented in [40]. Moreover, [3] applies the RNN model into optimal resource allocation problems based on the work in [33]. The letter presents that the network can efficiently dispatch a set of emergency personnel to a given set of events (e.g. fire events or injurious people) to provide

services. The authors in [41] discuss the problem of assigning each task in a parallel program to some resources and the results show that the RNN can be easier to implement on a parallel machine and improve the speed of processing compared with other heuristic approaches.

## 3. The RNN Based Algorithm for Resource Allocation

During an ongoing hazard, although the rescuers can identify the location of victims with the help of the DBES platform, reach the injured sites quickly, and wear protective devices, it is inappropriate to assume that the rescuers can always success in executing a mission [42] and it is uncertain whether the rescuer will reach the hazardous areas and carry the trapped people to the exits. Also, the decision-making of the assignment of rescuers to victims should be fast since long exposure to the hazards means less chance to survive [1]. Therefore, the section introduces an RNN based algorithm to provide near-optimal advice in real-time in uncertain environments. The mathematical model and the most important properties of the RNN is introduced firstly, and then we describe a task assignment algorithm based on the RNN techniques.

### *3.1. The Random Neural Network Model*

The RNN with synchronized interactions can solve the optimization problem of allocating a set of tasks to a limited set of resource simultaneously very quickly and quite accurately, and the basic ideas for the RNN model purposed in [3] are shown below.

The main symbols used for the RNN model are summarised in Table 1, and an application of one earlier result (Theorem 1) proposed in [43] is being used.

**Table 1.** List of the RNN training model symbols [3]

| Notation | Definition |
|---|---|
| $r_i$ | Firing rate for neuron $i$ |
| $k_i(t)$ | Internal state of neuron $i$ at time $t$ |
| $\Lambda(i)$ | Excitatory spike from the outside world to neuron $i$ |
| $\lambda(i)$ | Inhibitory spike from the outside world to neuron $i$ |
| $p^+(i,j)$ | Probability of excitatory spike from neuron $i$ to neuron $j$ |
| $p^-(i,j)$ | Probability of inhibitory spike from neuron $i$ to neuron $j$ |
| $d(i)$ | Probability of departure spike from neuron $i$ to the outside world |
| $Q(i,j,m)$ | Probability of neuron $i$ synchronous interaction together with neuron $j$ to affect third neuron $m$ |

**Theorem 1** *Let $\lambda^-(i)$ and $\lambda^+(i)$, $i = 1, ..., N$ be given by the following system of equations*

$$\lambda^-(i) = \lambda(i) + \sum_{j=1}^{N} r_j q_j \left[ p^-(j,i) + \sum_{m=1}^{N} Q(j,i,m) \right] \tag{1}$$

$$\lambda^+(i) = \sum_{j=1}^{N} r_j q_j p^+(j,i) + \sum_{j=1}^{N}\sum_{m=1}^{N} q_j q_m r_j Q(j,m,i) + \Lambda(i), \tag{2}$$

*where*

$$q_i = \lambda^+(i)/(r_i + \lambda^-(i)). \tag{3}$$

*If a unique nonnegative solution* $\{\lambda^-(i), \lambda^+(i)\}$ *exists for the nonlinear system of Equations 1 to 3, such that* $q_i < 1 \ \forall i$ *then*

$$\pi(\underline{k}) = \prod_{i=1}^{N} (1-q_i) q_i^{k_i}. \tag{4}$$

The network will have a solution if $q_i < 1$: it guarantees that the total arrival rate is less than the total departure rate so that the network will be finally in steady-state. According to the definition of $p^+(i,j)$, $p^-(i,j)$, $d(i)$, and $Q(i,j,m)$ in Table 1, we have a relationship between four different types of probabilities.

$$d(i) = 1 - \sum_{j=1}^{N} \left[ p^+(i,j) + p^-(i,j) + \sum_{m=1}^{N} Q(i,j,m) \right]. \tag{5}$$

We now use notation "weight" [44] to represent the rates at which the neurons interact [3], and the weights are defined as a simple product form of the firing rate and its corresponding probability.

$$w^+(i,j) = r_i p^+(i,j), \tag{6}$$

$$w^-(i,j) = r_i p^-(i,j), \tag{7}$$

$$w(i,j,l) = r_i Q(i,j,l). \tag{8}$$

Combine Equations 5 to 8, we can rewrite Equation 5:

$$\begin{aligned} r_i &= \frac{r_i \sum_{j=1}^{N} \left[ p^+(i,j) + p^-(i,j) + \sum_{m=1}^{N} Q(i,j,m) \right]}{1 - d(i)} \\ r_i &= \frac{\sum_{j=1}^{N} \left[ w^+(i,j) + w^-(i,j) + \sum_{m=1}^{N} w(i,j,m) \right]}{1 - d(i)}. \end{aligned} \tag{9}$$

The denominator of $q_i$ in Equation 3 can then be rewritten combining with Equations 1, 7, and 8:

$$\begin{aligned} D(i) &= r_i + \lambda^-(i) \\ D(i) &= r_i + \lambda(i) + \sum_{j=1}^{N} r_j q_j \left[ p^-(j,i) + \sum_{m=1}^{N} Q(j,i,m) \right] \\ D(i) &= r_i + \lambda(i) + \sum_{j=1}^{N} q_j \left[ r_j p^-(j,i) + \sum_{m=1}^{N} r_j Q(j,i,m) \right] \\ D(i) &= r_i + \lambda(i) + \sum_{j=1}^{N} q_j \left[ w^-(j,i) + \sum_{m=1}^{N} w(j,i,m) \right], \end{aligned} \tag{10}$$

163 similarly its numerator becomes, if combining with Equations 2, 6, and 8:

$$\begin{aligned} N(i) &= \lambda^{+}(i) \\ N(i) &= \sum_{j=1}^{N} r_j q_j p^{+}(j,i) + \sum_{j=1}^{N}\sum_{m=1}^{N} q_j q_m r_j Q(j,m,i) + \Lambda(i) \\ N(i) &= \sum_{j=1}^{N} q_j w^{+}(j,i) + \sum_{j=1}^{N}\sum_{m=1}^{N} q_j q_m w(j,m,i) + \Lambda(i), \end{aligned} \tag{11}$$

$$so\ that \quad q_i = N(i)/D(i). \tag{12}$$

164 The results can be used to design an efficient learning algorithm by updating weights of all neurons
165 and then selecting a neuron which has the largest $q$.

166 *3.2. Task Assignment Algorithm Based on the RNN Model*

Consider a group of rescuers $R$ waiting at the building exits and some victims $V$ being trapped in the hazardous areas, and assume that the initial locations of both rescuers and injured civilians are known with the support of the DBES platform. According to the work in [26], the RNN parameters can be defined as, a cost $C(r,v)$ associated with each possible assignment of rescuer $r$ to victim $v$, a fail execution probability $q(r,v)$ representing that rescuer $r$ is unable to rescue victim $v$ while the fact that the rescuer has been assigned to the victim, and a penalty $K(v)$ for not executing rescue for victim $v$. In this case, any assignment of rescuers to victims will have an expected cost and the objective of the algorithm is to minimize the overall expense by selecting the optimal assignments in order to increase the number of injured civilians collected and reduce the total rescue time. We represent the decision of dispatching rescuer $r$ to save victim $v$ by the probability $p(r,v)$, and $p(r,v) = 1$ if rescuer $r$ is allocated to victim $v$ and $p(r,v) = 0$ if not [26]. Hence,

$$p(r,v) \in \{0,1\},\ and\ \sum_{v\in V} p(r,v) = \pi(r) \in \{0,1\},\ r \in R, v \in V, \tag{13}$$

167 where as $p(r,v)$ are either 0 or 1, $\pi(r) \in \{0,1\}$ represents that (a) $\pi(r) = 0$, rescuer $r$ will not be
168 assigned a task if the assignments related to rescuer $r$ increases the overall cost, and (b) $\pi(r) = 1$,
169 a rescuer cannot be re-assigned some other victims if the rescuer already has task (i.e. the rescuers
170 are non-reusable).

171 The number of victims and rescuers are denoted as $|V|$ and $|R|$, respectively. Based on the
172 RNN parameters $C(r,v)$, $K(v)$, $q(r,v)$, and $p(r,v)$, the objective function that we try to minimize
173 can be written as [26]:

$$Minimize\ C = \sum_{v\in V}\sum_{r\in R} C(r,v)p(r,v) + \sum_{v\in V} K(v)\prod_{r\in R}\{1-(1-q(r,v))p(r,v)\} \tag{14}$$

$$Subject\ to: (1)\sum_{v\in V} p(r,v) = \pi(r) \in \{0,1\},\ r \in R, \quad (2) p(r,v) \in \{0,1\},\ r \in R, v \in V,$$

where the first term of Equation 14 is the average cost used by the rescuers, and the second term represents the accumulative average cost of failing in executing rescue for each victim. For each possible task assignment, we assume that there are only two results, success or failure, and Equation 14 takes both the cost of succeeding in rescue and the penalty of failing in save into account. Thus, the addition of the first term and the second term indicates that the minimization of Equation 14 is to find a optimal balance between the successful execution costs and the fail rescue penalties. The first constraint ensures that we can allocate one particular rescuer to at most one victim, and the total number of allocations can be less than $|R|$ [26]. The second one assumes that $p(r,v) = 1$ if rescuer $r$ is allocated to victim $v$ and $p(r,v) = 0$ if not. Since these two constraints, the expected failure probability $\{1-(1-q(r,v))p(r,v)\}$ [26] can be rewritten in an exponential form,

$$\begin{aligned} F &= 1-(1-q(r,v))p(r,v) = 1-(1-q(r,v))(p(r,v)=0 \; or \; 1) \\ F &= 1, \; if \; p(r,v)=0 \;\; or \;\; q(r,v), \; if \; p(r,v)=1 \\ Thus, \; F &= q(r,v)^{p(r,v)}. \end{aligned} \tag{15}$$

Substitue Equation 15 into Equation 14, we obtain,

$$Minimize \; C = \sum_{v\in V}\sum_{r\in R} C(r,v)p(r,v) + \sum_{v\in V} K(v)\prod_{r\in R} q(r,v)^{p(r,v)} \tag{16}$$

$$Subject \; to: (1)\sum_{v\in V} p(r,v) = \pi(r) \in \{0,1\}, \; r\in R, \quad (2)p(r,v)\in\{0,1\}, \; r\in R, v\in V.$$

In order to find the minimum solution of Equation 16, it is possible to enumerate all possible allocations of rescuers to victims, and then select the optimal result. Because of the high computational complexity, the RNN algorithm described before is used to solve Equation 16, which can provide a quick and accurate solution. Here, each possible assignment decision $(r,v)$ is represented by a neuron $N(r,v)$, and a list of symbols used for the approach is summarized in Table 2. The excitatory and inhibitory signals' arrival rates of each neuron $N(r,v)$ are specified so that the RNN can be used for the heuristic solution to optimization problems [26]. Hence, we represent the external rate of excitation and inhibition as

$$\Lambda(r,v) = max\{0, b(r,v)\} \qquad \lambda(r,v) = max\{0, -b(r,v)\}$$
$$\text{where } b(r,v) = K(v)(1-q(r,v)) - C(r,v),$$

where $K(v)(1-q(r,v))$ is the expected reduction in the penalty of not rescuing victim $v$ if rescuer $r$ has been dispatched to save it, and $C(r,v)$ is the cost of triggering rescuer $r$ to save victim $v$. $b(r,v)$ can be viewed as a characteristic of the neuron $(r,v)$, where $b(r,v) > 0$ if this allocation could minimize the overall cost of task assignments and $b(r,v) \leq 0$ if not. That is, if the expected reduction in the cost of not saving victim $v$ is larger than the cost of allocating rescuer $r$ to victim $v$ (i.e. $b(r,v) > 0$) then we think this assignment is efficient, while if not then this allocation will not be executed. Moreover, in order to avoid resource wastage [1] we discourage the allocation of more than one rescuers to the same victim, thus, we set the inhibitory weights [26]:

**Table 2.** List of RNN parameter association approach symbols [26]

| Notation | Definition |
|---|---|
| $K(v)$ | Penalty for not rescuing victim $v$ |
| $q(r,v)$ | Probability rescuer $r$ is unable to rescue victim $v$ |
| $C(r,v)$ | Cost for saving victim $v$ by rescuer $r$ |
| $\Lambda(r,v)$ | External arrival rate of excitatory signals to neuron $(r, v)$ |
| $\lambda(r,v)$ | External arrival rate of inhibitory signals to neuron $(r, v)$ |
| $w^+(r,v;r',v')$ | Rate of excitatory signals to neuron $(r, v)$ from firing neuron $(r', v')$ |
| $w^-(r,v;r',v')$ | Rate of inhibitory signals to neuron $(r, v)$ from firing neuron $(r', v')$ |
| $r(r,v)$ | Firing rate of neuron $(r, v)$ |
| $Q(r,v)$ | Probability neuron $(r, v)$ is excited |

$$w^-(r,v;r',v) = max\{0, b(r,v)\}, \quad if\ r \neq r',$$

which indicates that if the allocation of rescuer $r$ to victim $v$ is able to minimize the objective function (i.e. $b(r,v) > 0$), the allocations of other rescuers to the same victim should be discouraged, otherwise ($b(r,v) \leq 0$), this allocation should have no effect on other assignments ($w^-(r,v;r',v) = 0$). Similarly we wish to avoid assigning more than one victims to the same rescuer:

$$w^-(r,v;r,v') = max\{0, b(r,v)\}, \quad if\ v \neq v'.$$

For the sake of simplicity, we are not reinforcing or impairing the other assignments [26], so that $w^+(r,v;r,v') = 0$ and $w^-(r,v;r',v') = 0$ for all other $r$, $r'$ and $v$, $v'$, and we assume that the synchronous interaction between two neurons cannot affect some third neuron, i.e. $w(i,j,m) = 0$. Hence, we rewrite Equation 9,

$$r(r,v) = \frac{\sum_{r',v'}\left[w^+(r,v;r',v') + w^-(r,v;r',v')\right]}{1 - d(r,v)} = \sum_{r',v'} w^-(r,v;r',v'). \tag{17}$$

Equation 12 is then rewritten:

$$\begin{aligned} Q(r,v) &= \Lambda(r,v)/[\lambda(r,v) + r(r,v) + \sum_{r',v'} Q(r',v')w^-(r',v';r,v)] \\ Q(r,v) &= \Lambda(r,v)/[\lambda(r,v) + r(r,v) + \sum_{r'\neq r} Q(r',v)w^-(r',v;r,v) + \sum_{v'\neq v} Q(r,v')w^-(r,v';r,v)] \end{aligned} \tag{18}$$

When there are civilians who are trapped in the hazardous areas, the rescuers will be noticed by the sensor network in the DBES multi-agent environment. Next, Equation 18 is solved iteratively until convergence in the following manner (Algorithm 1) to obtain a optimal assignment of rescuers to victims. In the algorithm, $R$ is the original set of rescuers and $R_{rem}$ is the set of remaining rescuers, $S$ is the solution set where the assigned rescuer-victim pairs are stored [26], and $K_{cur}(v)$ is the current penalty of victim $v$, which will be updated if victim $v$ is allocated a rescuer.

Note that using this algorithm, the assignments in solution set $S$ always result in reducing the cost of the objective function [26], since an assignment may be selected if $Q(r,v) > 0$ or

**Algorithm 1** Task assignment for optimal emergency management

1: initialize $R_{rem} = R$; $S = \emptyset$; $K_{cur}(v) = K(v)\ v \forall V$; $q(r,v)\ r \forall R, v \forall V$; $C(r,v)\ r \forall R, v \forall V$; $Q_{pre}(r,v) = Q(r,v) = 0\ \ r \forall R, v \forall V$;
2: **for** each possible assignment pairs $(r,v) \in [neurons]$ **do**
3: $b(r,v) = K(v)(1 - q(r,v)) - C(r,v)$;
4: $\Lambda(r,v) = max\{0, b(r,v)\}$;
5: $\lambda(r,v) = max\{0, -b(r,v)\}$;
6: $w^-(r,v;r',v) = max\{0, b(r,v)\}, \quad if\ r \neq r'$;
7: $w^-(r,v;r,v') = max\{0, b(r,v)\}, \quad if\ v \neq v'$;
8: **end for**
9: **for** each possible assignment pairs $(r,v) \in [neurons]$ **do**
10: $r(r,v) = \sum_{r',v'} w^-(r,v;r',v')$;
11: **end for**
12: **for** each possible assignment pairs $(r,v) \in [neurons]$ **do**
13: $Q_{pre}(r,v) = Q(r,v)$;
14: $Q(r,v) = \Lambda(r,v)/[\lambda(r,v) + r(r,v) + \sum_{r' \neq r} Q(r',v) w^-(r',v;r,v) + \sum_{v' \neq v} Q(r,v') w^-(r,v';r,v)]$;
15: **end for**
16: initialise $convergence = true$;
17: **for** each possible assignment pairs $(r,v) \in [neurons]$ **do**
18: **if** $Q_{pre}(r,v) \neq Q(r,v)$ **then**
19: $convergence = false$;
20: break;
21: **end if**
22: **end for**
23: **if** *convergence* **then**
24: // select the most excited neruon (go to line 28);
25: **else**
26: // continue iterating until convergence (go to line 12);
27: **end if**
28: initialize maxValue = 0;
29: **for** each possible assignment pairs $(r^*, v^*) \in [neurons]$ **do**
30: **if** $Q(r^*, v^*) > maxValue$ **then**
31: $maxValue = Q(r^*, v^*)$;
32: select rescuer-victim pair $(r^*, v^*)$;
33: **end if**
34: **end for**
35: update $S = S \bigcup (r^*, v^*)$, $R_{rem} = R_{rem} \setminus r^*$, $K_{cur}(v^*) = K_{cur}(v^*) q(r^*, v^*)$;
36: **if** $R_{rem}$ is not empty **then**
37: // allocate next rescuer (go to line 2);
38: **else**
39: stop;
40: **end if**

[1]

$\Lambda(r,v) > 0$ (see Line 30) which means $b(r,v) > 0$, otherwise if $b(r,v) \leq 0$ then $Q(r,v) = \Lambda(r,v) = 0$, the allocation will not be executed. The penalty of not saving victim $v$ is updated by $K_{cur}(v^*) = K_{cur}(v^*)q(r^*,v^*)$ (see Line 35), since $K_{cur}(v^*) = K_{cur}(v^*) - K_{cur}(v^*)(1 - q(r^*,v^*))$ $(i.e.\ expected\ reduction\ on\ penalty) = K_{cur}(v^*)q(r^*,v^*)$.

## 4. Evaluation of the Task Assignment Algorithm

In this section, we begin by presenting the assumptions we made for the simulations, and then we describe the simulation model, finally the results will be illustrated.

### *4.1. Assumptions*

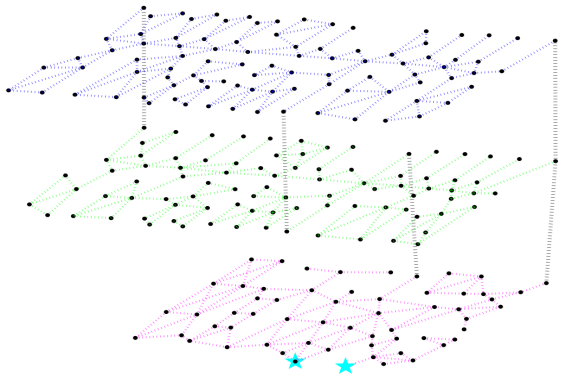
The first assumption is that the initial health status of civilians and rescuers is 100, and the health will be affected by the hazards, i.e. their health level decreases gradually when they are exposed to hazards. If the remaining health of civilians is less than a threshold, they are assumed to be immobilized. These victims are referred to as tasks and they need to be assigned to the rescuers. Additionally, in real emergencies the rescuers will wear protective clothes, thus, we assume that the health level of rescuers decreases slower than that of civilians. Another assumption is that the capacity of each rescuer is one and the maximum capacity of each of them is two, which means that each rescuer will be allocated one trapped civilian each time but it allows for the possibility that one more trapped civilian is assigned to them if the location of the civilian is near the way selected by the rescuer towards the exits. Finally, the rescuers are assumed to be non-usable so that they will execute tasks only once.

### *4.2. Description of the Simulation Model*

The simulation model is established based on the Distributed Building Evacuation Simulator [11], where all simulated entities including evacuees as well as rescuers are modelled as agents, and these entities are able to communicate with each other. We assume that this is a fire-related emergency evacuation and the area to evacuate is a building based on the three lower floors of Imperial College London's EEE building [14]. A coarse graph representation of this building is showed in Figure 1.

The civilians' goal is to get out the hazards along with a safe path to avoid injury, but if they are at stake, they will send a message including ID, location, and health status to the rescuers and ask for help, while the rescuers aim to get into the hazardous areas to carry victims to the safe sites. When the evacuees need help in the building, the RNN based algorithm computes an optimum resource allocation scheme to dispatch a limited set of rescuers to the location of victims to provide support. The navigation algorithm used by the rescuers is considered separately for two stages of the rescue process, moving to the injured civilians and carrying them to the exits. In the first stage, the rescuers try to arrive at the emergency sites quickly to assist the injured civilians in real-time, and we assume that they cannot be hurt by fire when they get through the hazards themselves since they wear protective clothes. Therefore, the dominant factor for the navigation

**Figure 1.** Graph representation of the building model [14]. The two cyan stars on the ground floor mark the position of the building's exits.

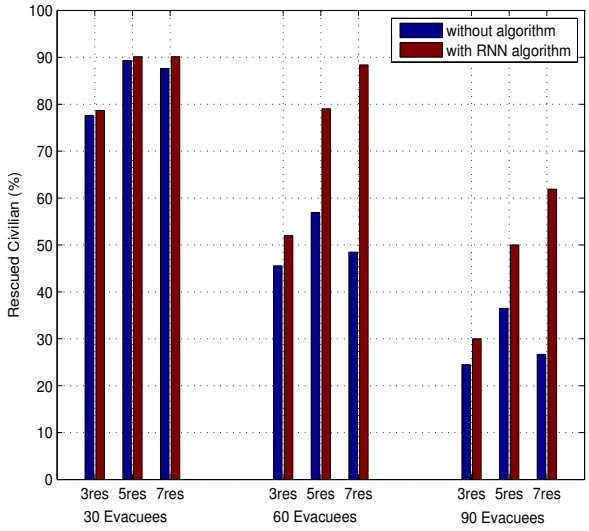


algorithm is the time, i.e. the algorithm should guide the rescuers to the injured locations as quick as possible. We choose to use the Dijkstra's Shortest Path Algorithm which can compute the shortest path between two nodes based on the physical length of every link between them, and here the shortest path is regarded as the quickest path since the moving speed of rescuers is assumed to be fixed. However, the rescuers take injured people to the exits in the second stage, so that the rescuers should not only go to the exits rapidly but protect people from further injuring. Thus, for the sake of simplicity we still use the Dijkstra's Algorithm, but the length between two nodes is computed based on its physical distance as well as the hazard intensity on the link connecting them [2]. Thus, the path selected by the rescuers can navigate them to the evacuation exits rapidly and take them away from the hazards to keep people safe.

### *4.3. Performance Evaluation*

The effectiveness of the proposed RNN based task assignment algorithm is evaluated in terms of two performance metrics, (1) percentage of evacuees who are saved by rescuers based on the total number of injured civilians; (2) number of victims saved by rescuers with the sizes of the set of rescuers and the set of victims given, and both the two metrics denote straightforwardly the efficiency of the RNN based algorithm. The results of our experiments are obtained by comparing two different scenarios, without task assignment algorithm and with the RNN based algorithm, under diverse sizes of the set of rescuers and the set of victims. Each result is obtained by executing 10 simulation runs, and for each simulation run, the locations of injured civilians are randomized around the hazards in the building. The purpose of the simulations is to check if the RNN based algorithm can avoid the waste of resources and reduce the inefficiency of resource allocations. Thus, to keep things as simple as possible the cost of each assignment $C(r, v)$ and the fail rescue probability $q(r, v)$ are taken to be independent from its assigned victim $v$ [26] and generated randomly. Here, we do not address how to compute an appropriate execution cost and fail probability according to the interaction between a rescuer and a victim, which will be considered in future work. Specifically, $K(v)$, $C(r)$, and $q(r)$ are generated from the uniform distribution in the interval $[0, 50]$, $[0, 10]$, and $[0.05, 0.15]$ respectively, and $Q(r, v)$ is initialized to zero for all assignments. All RNN parameters are updated 20 times for each allocation to reach

**Figure 2.** Percentage of evacuees saved by rescuers based on the total number of injured civilians.




convergence, and the assignment with the highest probability of being active (i.e. $Q(r, v)$) will be selected. The results are summarized in Table 3 and Table 4.

**Table 3.** Percentage of evacuees saved by rescuers based on the total number of injured civilians

| No. of civilians | No. of rescuers | Without algorithm | With RNN algorithm |
|---|---|---|---|
| 30 | 3 | 77.62% | 78.67% |
| 30 | 5 | 89.29% | 90.14% |
| 30 | 7 | 87.62% | 90.14% |
| 60 | 3 | 45.56% | 51.99% |
| 60 | 5 | 56.91% | 79.03% |
| 60 | 7 | 48.46% | 88.38% |
| 90 | 3 | 24.50% | 30.00% |
| 90 | 5 | 36.48% | 50.00% |
| 90 | 7 | 26.66% | 61.90% |

Table 3 shows that the rescue system with the RNN based task assignment algorithm can increase the percentage of evacuees saved by rescuers based on the total number of injured civilians, and Figure 2 represents the data obtained from Table 3 by using bar charts. In Figure 2, it is clear that the percentage of rescued civilians decreases with increasing number of evacuees in the building, and this is due to the fact that it takes relatively long time to evacuate in high population density environments because of congestion problems, which then increases the time of exposure to the hazards and reduces the number of survivors. Without the task assignment algorithm, the system with 7 rescuers reduces the percentage of rescued civilians compared with the system with 5 rescuers for each case (30, 60, and 90 evacuees), which is more obvious in high population

**Table 4.** Number of evacuees saved by rescuers

| **No. of rescuer $\|R\|$** | **No. of victims $\|V\|$** | **Without algorithm** | **With RNN algorithm** |
|---|---|---|---|
| 3 | 4 | 3.00 | 3.80 |
| 5 | 4 | 3.43 | 4.00 |
| 7 | 4 | 4.00 | 4.00 |
| 3 | 8 | 4.71 | 5.75 |
| 5 | 8 | 6.00 | 7.50 |
| 7 | 8 | 5.89 | 8.00 |
| 3 | 16 | 4.88 | 6.00 |
| 5 | 16 | 7.40 | 9.75 |
| 7 | 16 | 6.67 | 13.25 |

densities (60 and 90 evacuees). The reason is that when there are people who need help, the rescuers enter the building and move towards the hazardous areas, so that some of them may increase the congestion along with the road [25]. Hence the rescuers and the remaining evacuees will probably spend long time waiting or be obstructed by the congestion. Moreover, it is possible that (1) different rescuers are allocated to the same victim, (2) different victims are assigned to the same rescuer, and (3) a rescuer may be injured during rescue and then cannot finish the task. The reason for phenomenons (1) and (2) is that there is no relationship and effect between each assignments, that is, the allocation of a rescuer to a victim does not consider the assignments that have already been done. In high population density environments, more evacuees may be at stake at bottleneck points [45], such as stairs and evacuation exits, due to the pedestrian congestion, and then more people need to be assigned a rescuer simultaneously. Therefore, it is more likely that the rescuers receive more than one rescue messages from the sensor network and/or the same rescue message is sent to more than one rescuers. In this case, the rescuers select their target randomly, which leads to problems (1) and (2). The result of (3) indicates that the rescuer may be not perfectly suited to save a victim though the rescuer has been allocated [26]. Because the rescuers may not know the inside situations in uncertain environments, and they may be exposed to highly hazardous areas long time since they move to the disasters and may be obstructed by the congestion. Therefore, if introducing more rescuers in the system where does not apply the task assignment algorithm, it will not only aggravate the congestion, but also may not increase the number of people rescued due to problems (1) to (3). In this case, the performance of the system with more rescuers may be worse.

However, with the RNN based algorithm, the percentage of rescued civilian has a tendency to increase with more rescuers in the system (see Figure 2). Although the rescuers still increase the congestion in the building, the RNN based algorithm makes the rescuers save victims effectively and quickly, which dominates the overall performance of the rescue system. This is because the RNN based algorithm can alleviate the three problems mentioned above by modelling $N$ fully connected neurons network with a cost objective function. The algorithm treats each rescuer-victim pair as a neuron, such as neuron $i$ means that rescuer $r$ is allocated to victim $v$, and it can

minimize the total cost by solving the objective function. The objective function takes into account the cost of allocating rescuer $r$ to victim $v$, the probability that rescuer $r$ is unable to save victim $v$ (say, fail rescue probability of neuron $i$), and the penalty of not rescuing victim $v$. Therefore a rescuer-victim pair will be selected if the allocation can minimize the objective function, for example, if the fail rescue probability of neuron $i$ is high, the neuron may not be selected since it might increase the overall cost. Hence, the problem (3) is solved. Moreover, the algorithm considers the rate of excitatory and inhibitory signals from firing neuron $i$ to neuron $j$. If neuron $i$ has been selected, the inhibitory signals from neuron $i$ to other neurons $j$ where $j \neq i$ will be updated to discourage the allocations of different rescuers to the same victim and different victims to the same rescuer. Then the problems (1) and (2) can be avoided to some degree. Therefore, more rescuers are able to save more people with the RNN based algorithm, and the percentage of rescued civilians can be increased. Comparing the two scenarios, without and with the task assignment algorithm, Figure 2 shows that the system with the RNN based algorithm performs better, especially in high population density environments, which accords with the analyses above.

In Table 4, given a set of victims, the increase of the number of rescuers cannot significantly improve the performance on the number of rescued evacuees without the task assignment algorithm, and more rescuers may reduce the number of rescued evacuees, which agrees with the results obtained from Figure 2. However, the system with the RNN based algorithm can provide a near-optimal performance, where:

$$
\begin{aligned}
&If \ \ |V| < overall\ maximum\ capacity\ of\ rescuers \\
&\qquad number\ of\ rescued\ evacuees \overset{approx}{\longrightarrow} |V| \\
&else \\
&\qquad number\ of\ rescued\ evacuees \overset{approx}{\longrightarrow} overall\ maximum\ capacity\ of\ rescuers \\
&end.
\end{aligned}
$$

Here, the capacity of rescuers is one and the maximum capacity of rescuers is two, which has been assumed in Section 3. Note that, if the injured civilians are far away from each other, the number of rescued civilians for each rescuer may not reach the maximum capacity. This is due to the assumption that the capacity of each rescuer is one, but only if there is a victim who is near the way that the rescuer moves along with towards exits and the capacity of the rescuer does not reach two (i.e. the maximum capacity), this victim will be carried. Therefore, if the locations of the injured civilians are not close, it is impossible to select a route by the rescuer that will get through other injured locations.

## 5. Conclusions and Future Work

In the paper, we propose a task assignment algorithm based on the RNN techniques which can efficiently dispatch rescuers to aid victims to decrease the fatalities in emergency situations with considering the uncertainty of dynamic hazard environments. The simulations show that the algorithm can provide a approximate real-time allocation of rescuers to victims. Also the allocation solutions are close to the optimal ones where if the number of injured civilians is less

than the total number of people that the rescuers can save, all injured civilians could be rescued by the rescuers, and if not, the number of saved civilians could reach the overall maximum capacity of the rescuers.

In future work, the task assignment algorithm can be evaluated with more constraints, such as the cost of each assignment $C(r, v)$ and the fail rescue probability $q(r, v)$ can be generated relying on the situations of both the rescuers and victims. Our simulations take these two parameters to be independent from its assigned victim and initialize them randomly, but it may be better if we generate them considering the associated victim. For example, for each rescuer-victim pair, the distance between them can be computed with the help of the DBES platform and the hazard intensity around the victim also can be estimated, and then the cost of assignment and the fail execution probability can be evaluated with taking these two factors into account. In this way, we can obtain a more precise prediction of the expense of the allocation and make a more optimal assignment. Additionally, when there are fewer rescuers than the injured civilians, an efficient clustering scheme [24] can be applied to group victims according to their locations so that the rescuer can provide support to as many civilians as possible, such as delivering protective devices, though they may be cannot carry all of them out of the building. What's more, the congestion problem caused by the rescuers should be alleviated and the navigation approach with the CPN can be applied in rescue process, which is able to provide a faster optimal path finding and cause less congestion compared with the Dijkstra's algorithm [14].

## References


1. S. Li, A. Zhan, X. Wu, P. Yang, and G. Chen, "Efficient emergency rescue navigation with wireless sensor networks," *Journal of Information Science and Engineering*, vol. 27, no. 1, pp. 51–64, 2011.
2. A. Filippoupolitis and E. Gelenbe, "A distributed decision support system for building evacuation," in *Human System Interactions, 2009. HSI'09. 2nd Conference on*, 2009, pp. 323–330.
3. E. Gelenbe and S. Timotheou, "Random neural networks with synchronised interactions," *Neural Computation*, vol. 20, no. 9, pp. 2308–2324, Jul. 2008.
4. D. W. Pentico, "Assignment problems: A golden anniversary survey," *European Journal of Operational Research*, vol. 176, pp. 774–793, Jan. 2007.
5. A. Filippoupolitis, "Emergency simulation and decision support algorithms," *Thesis submitted for the Degree of Doctor of Philosophy of the University of London and the Diploma of Imperial College*, Oct. 2010.
6. Y.-C. Tseng, M.-S. Pan, and Y.-Y. Tsai, "Wireless sensor networks for emergency navigation," *IEEE Computer*, vol. 39, no. 7, pp. 55–62, Jul. 2006.
7. M.-S. Pan, C.-H. Tsai, and Y.-C. Tseng, "Emergency guiding and monitoring applications in indoor 3D environments by wireless sensor networks," *Int'l J. Sensor Networks*, vol. 1, no. 1/2, pp. 2–10, Sept. 2006.
8. M. Barnes, H. Leather, and D. K. Arvind, "Emergency evacuation using wireless sensor networks," in *IEEE Conf. Local Computer Networks*, 2007, pp. 851–857.

9. T. Tabirca, K. N. Brown, and C. J. Sreenan, "A dynamic model for fire emergency evacuation based on wireless sensor networks," *Parallel and Distributed Computing*, pp. 29–36, 2009.
10. A. Filippoupolitis, E. Gelenbe, D. Gianni, L. Hey, G. Loukas, and S. Timotheou, "Distributed agent-based building evacuation simulator," in *Summer Computer Simulation Conference*, 2008.
11. N. Dimakis, A. Filippoupolitis, and E. Gelenbe, "Distributed building evacuation simulator for smart emergency management," *The Computer Journal*, vol. 53, no. 9, pp. 1384–1400, Feb. 2010.
12. J. Smith, "State-dependent queueing models in emergency evacuation networks," *Transportation Research Part B: Methodology*, vol. 25, no. 6, pp. 373–389, Feb. 1991.
13. A. Desmet and E. Gelenbe, "Graph and analytical models for emergency evacuation," *Future Internet*, vol. 5, no. 1, pp. 46–55, 2013.
14. H. Bi, A. Desmet, and E. Gelenbe, "Routing emergency evacuees with cognitive packet networks," in *Proceedings 28th Annual International Symposium on Computer and Information Sciences ISCIS 2013*. Springer, 2013.
15. E. Gelenbe, "Cognitive packet network," *U.S. Patent 6,804,201*, Oct. 11 2004.
16. E. Gelenbe, R. Lent, and A. Nunez, "Self-aware networks and qos," *Proceedings of the IEEE*, vol. 92, no. 9, pp. 1478–1489, Sept. 2004.
17. E. Gelenbe, "Steps toward self-aware networks," *Communications of the ACM*, vol. 52, no. 7, pp. 66–75, Feb. 2009.
18. G. Gorbil and E. Gelenbe, "Opportunistic communications for emergency support systems," in *Int'l Conf. Ambient Systems, Networks and Technologies*, 2011, pp. 1–9.
19. G. Loukas, S. Timotheou, and E. Gelenbe, "Robotic wireless network connection of civilians for emergency response operations," in *Int'l Symp. Computer and Information Sciences*, 2008, pp. 1–6.
20. J. Reich and E. Sklar, "Robot-sensor networks for search and rescue," in *IEEE Int'l Workshop Safety, Security and Rescue Robotics*, 2006.
21. X. Li, N. Santoro, and I. Stojmenovic, "Localized distance-sensitive service discovery in wireless sensor and actor networks," *IEEE Trans. Computers*, vol. 58, no. 9, pp. 1275–1288, Sept. 2009.
22. A. Verma, H. Sawant, and J. Tan, "Selection and navigation of mobile sensor nodes using a sensor network," *Pervasive and Mobile Computing*, vol. 2, no. 1, pp. 65–84, 2006.
23. I. Stojmenovic, D. Liu, and X. Jia., "A scalable quorum based location service in ad hoc and sensor networks," *Int'l J. Comm. Networks and Distributed Systems*, vol. 1, no. 1, pp. 71–94, 2008.
24. Y.-C. Wang, W.-C. Peng, M.-H. Chang, and Y.-C. Tseng, "Exploring load-balance to dispatch mobile sensors in wireless sensor networks," *IEEE Int'l Conf. Computer Comm. and Networks*, pp. 669–674, Aug. 2007.
25. S. Li, A. Zhan, X. Wu, and G. Chen, "Ern: Emergence rescue navigation with wireless sensor networks," *Parallel and Distributed Systems*, pp. 361–368, 2009.

26. E. Gelenbe, S. Timotheou, and D. Nicholson, "Fast distributed near-optimum assignment of assets to tasks," *The Computer Journal*, vol. 53, no. 9, pp. 1360–1369, 2010.
27. E. Gelenbe, "Random neural networks with negative and positive signals and product form solution," *Neural Computation*, vol. 1, no. 4, pp. 502–510, 1989.
28. ——, "Stability of the random neural network model," *Neural Computation*, vol. 2, no. 2, pp. 239–247, 1990.
29. E. Gelenbe, Z. Mao, and Y. Li, "Function approximation with random neural network," *IEEE Trans. Neural Networks*, vol. 10, no. 1, Jan. 1999.
30. ——, "Function approximation by random neural networks with a bounded number of layers," *Journal of Differential Equations and Dynamical Systems*, vol. 12, pp. 143–170, Feb. 2004.
31. E. Gelenbe and J.-M. Fourneau, "Random neural networks with multiple classes of signals," *Neural Computation*, vol. 11, no. 4, pp. 953–963, May 1999.
32. E. Gelenbe and K. F. Hussain, "Learning in the multiple class random neural network," *IEEE Trans. on Neural Networks*, vol. 13, no. 6, pp. 1257–1267, Nov. 2002.
33. E. Gelenbe and S. Timotheou, "Synchronized interactions in spiked neuronal networks," *The Computer Journal*, vol. 51, no. 6, pp. 723–730, Mar. 2008.
34. E. Gelenbe, M. Sungur, C. Cramer, and P. Gelenbe, "Traffic and video quality with adaptive neural compression," *Multimedia Systems*, vol. 4, no. 6, pp. 357–369, Dec. 1996.
35. E. Gelenbe, Y. Feng, and K. R.Krishnan, "Neural network methods for volumetric magnetic resonance imaging of the human brain," *Proceedings of the IEEE*, vol. 84, no. 10, pp. 1488–1496, Oct. 1996.
36. E. Gelenbe and G. Sakellari, "Admission of qos aware users in a smart network," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 3, no. 1, pp. 14–28, Mar. 2008.
37. E. Gelenbe and G. Loukas, "A self-aware approach to denial of service defence," *Computer Networks*, vol. 51, no. 5, pp. 1299–1314, Apr. 2007.
38. E. Gelenbe, P. Liu, and J. Laine, "Genetic algorithms for route discovery," *IEEE Transactions on Systems, Man and Cybernetics B*, vol. 36, no. 6, pp. 1247–1254, Dec. 2006.
39. E. Gelenbe, V. Koubi, and F. Pekergin, "Dynamical random neural network approach to the travelling salesman problem," *Proc. IEEE Symp. Syst., Man, Cybern.*, vol. 2, pp. 630–635, Oct. 1993.
40. E. Gelenbe, A. Ghanwani, and V. Srinivasan, "Improved neural heuristics for multicast routing," *IEEE J. Selected Areas in Communications*, vol. 15, no. 2, pp. 147–155, Feb. 1997.
41. J. Aguilar and E. Gelenbe, "Task assignment and transaction clustering heuristics for distributed systems," *Information Sciences*, vol. 97, no. 1-2, pp. 199–219, Mar. 1997.
42. E. Gelenbe and F.-J. Wu, "Large scale simulation for human evacuation and rescue," *Computers and Mathematics with Applications*, vol. 64, no. 12, pp. 3869–3880, December 2012.
43. E. Gelenbe, "G-networks with triggered customer movement," *Applied Probability*, vol. 30, no. 3, pp. 742–748, Sept. 1993.
44. ——, "Learning in the recurrent random neural network," *Neural Computation*, vol. 5, no. 1, pp. 154–164, Jan. 1993.

45. E. Gelenbe and G. Görbil, "Opportunistic communications for emergency support systems," *Procedia Computer Science*, vol. 5, pp. 39–47, 2011.