Manage Crowds and Vehicles in an Emergency

by
Qing Han

Supervisor: Professor Erol Gelenbe

A Thesis submitted in fulfilment of requirements for the degree of
Master of Science
Communications and Signal Processing
of Imperial College London

Department of Electrical and Electronic Engineering
Imperial College London
August 13, 2013

Abstract

Managing crowds effectively in emergencies requires emergency personnel to provide support to injured evacuees who are obstructed by the hazard, such as fire and smokes. During an ongoing hazard, the unprotected evacuees may be trapped and find no way out due to fire or other obstacles, and consequently they have to wait for rescuers to remove obstacles and take them out. Furthermore, the rescuers can take actions strategically in order to save people out of danger maximally, efficiently, and quickly. This report studies the effectiveness of rescuers who search for injured evacuees and carry them to the exits, and also considers task assignment problem involving optimally matching rescuers and victims so that rescuers can assist trapped civilians in real time. The system is evaluated in various emergency scenarios, using the DBES multi-agent simulation platform for Building Evacuation. The simulations indicate that the presence of rescuers can result in a significantly higher percentage of safely evacuated people and a lower injury level of the evacuees. Additionally, Random Neural Network (RNN) based task assignment algorithm provides a near-optimal solution to resource allocation problems, which avoids resource wastage and improves the efficiency of rescue process.

Acknowledgment

Acknowledgments can be written here.

Contents

List of Figures

List of Tables

Abbreviations

DBES: Distributed Building Evacuation Simulator

DES: Discrete-Event Simulator

CPN: Cognitive Packet Network

oppcomms: Opportunistic Communications

ESS: Emergency Support System

RNN: Random Neural Network

WSN: Wireless Sensor Network

TORA: Temporally Ordered Routing Algorithm

GPA: Generalized Assignment Problem

Chapter 1

Introduction

Emergency evacuations [1, 2] create major technical issues of communications, distributed decision making and optimum planning and simulation which have become significant areas of research in their own right. The evacuation of a building is itself a challenging problem due to the high-intensity and fast-spreading of the hazard, which requires a highly reactive and adaptive system to provide dynamic and real-time solutions [3] which may be compounded by the sheer size of the building. The recent work [4] on this subject presents an adaptation of Cognitive Packet Network (CPN) [5–7], a Self-Aware computer network routing protocol, which is able to provide optimal and safe routes to evacuees in emergency situations. CPN is powerful enough to rapidly search for an optimal path within a large network with a changing environment. Compared with Dijkstra's Shortest Path Algorithm, CPN performs better with fast convergence time, less congestion and fatality rates, which effectively increase the efficiency and the success of the system in directing the evacuees towards the exits.

Also, previous work [8] has proposed the Opportunistic Communications (opp-comms) concept which can provide emergency support even when a stable communication infrastructure is unavailable. Oppcomms allows the exchange of packages at a close range of a few to some tens of meters with limited or no infrastructure and the hazard information is disseminated over an opportunistic network formed by communication nodes, thus, each civilian can be guided by his own communication node, which locally calculates the best evacuation path based on current information [9]. However, Oppcomms may be susceptibe to Denial of Service and other network attacks [10]. If used in the absence of network attacks, Oppcomms can support navigation for evacuation when the prior communication infrastructure fails or is unavailable, and will result in improved evacuation ratios even with high population densities.

The related work concentrates on successfully directing the evacuees towards evacuation exits along a safe path, however, to some extent there are still fatalities, and with the expansion of emergency areas and the increasing of population density, we should consider that some evacuees who are near the hazard are seriously injured and may be cannot move, and the hazard has affected all possible paths leading to them. What is worse is that even if people have at least one way out, navigating every civilian with the same strategy probably leads to congestion [11], which means that some of evacuees may spend lots of time waiting even though they know the evacuation path, and this is very dangerous in spreading disaster environments. As any spot may turn to dangerous at any time, longer waiting time means less chance to survive. In this case, only providing navigation information to direct civilians far away from hazardous areas to exits is not enough. Thus, the rescuer is necessary for emergency management, since rescue personnel can expose in the hazard up to a certain degree for a relative long time and provide support to potentially injured civilians. Nevertheless, although they could arrive at the hazardous sites quickly and protect people from hazard by using protective devices, they may not know the inside situations in dynamic disaster emergencies. For example, how the fire and smokes intensity, whether there is congestion, and which victim has been allocated a rescuer or which rescuer has been assigned a victim. These uncertainty factors may be lead to a waste of resources and the inefficiency of resource allocation, such as a rescuer is unable to rescue a victim, distinct rescuers are allocated to the same victim, and distinct victims are assigned to the same rescuer. Therefore a fast near-optimum assignment of rescuers to victims with a probabilistic outcome of success in rescuing should also be considered even though each rescuer can potentially save any of the victims.

A Random Neural Network (RNN) based task assignment approach can compute a fast approximate decision for a resource allocation that arises in emergency management for a problem whose exact solution would require an impractically large computation time [12]. There is a cost associated with each possible assignment of a rescuer to a victim, and if a victim does not being allocated a rescuer there is a penalty for not saving the victim, and there also is a fail execution probability representing that a rescuer is unable to save a victim despite the rescuer has been allocated to it. In RNN based approach, there is a cost objective function associated with RNN parameters, such as assignment cost, non-execution penalty, and fail execution probability, and the approach formulates the allocation of rescuers to victims in order to minimize the expected cost to avoid resource wastage and the inefficiency of resource allocation. By calculating RNN parameters iteratively until convergence, we can pick out a assignment which can maximally reduce the overall expected cost and then this assignment is executed. RNN based task assignment approach can provide fast and near-optimal resource allocation solutions even in high population density emergencies, and it can greatly improve the efficiency of emergency rescue. Therefore, rescuer, one of main force in emergency rescue, can make great contribution to saving people out of disasters with applying RNN based task assignment approach.

In order to make the simulation more realistic, firstly, civilian actors are modelled based on their age, divided into three categorises, adults, children and disabled, and they have different walking speeds, physiological capacity and fatigue status. The civilians' goal is to reach one of the evacuation exits with maximum health but if they are injured in the hazard and cannot move, they will send a message to rescuers and ask for help. In the multi-agent simulation environment, the smart space can capture the position and action of people in the area by using RFID devices and audiovisual sensors [13], thus the message received by rescuers can include the detailed information about the injured civilians, such as their ID, position, health level, and nature of injury. The duty of the rescuers is then to search for the injured civilians using the information received, and lead or take them them to the evacuation exits or assembly points. The rescuer is responsible to save people as much as possible if they are able to do so, e.g. if their health is still good enough, since we do not expect rescuers to sacrifice themselves during the emergencies. Moreover, in real emergency situations, the rescuers will wear protective clothes to be less susceptible to injuring themselves, and they will carry protective devices to assist injured civilians. Here, we model the functions of rescuers by making some assumptions that will be illustrated in Section 3.1.

The remainder of the report is organized as follows. Chapter 2 summarizes the related work on emergency evacuation systems and rescue systems. Chapter 3 describes the design details of the rescue system model, rescue process, and task assignment algorithm. Then the simulation results are presented in Chapter 4, in which we compare performances of the system without rescuers and the system with rescuers, and also show that RNN based task assignment algorithm can provide a near-optimal solution. Finally, in Chapter 5 we conclude the results and remaining problems of the report, and then we discuss open issues and suggest possible directions for future work.

Chapter 2

Background

In this report, we investigate the effect on introducing rescuers in building evacuation system when evacuees are injured and immobilitied, and discuss resource allocation problem for emergency management, i.e. how to assign the rescuers to the injured civilians in emergency situations. The use of rescuers presented in this report covers the areas of evacuation navigation and evacuation algorithm, distributed simulator, analytical performance model and dynamic task allocation algorithm. We discuss the related work in each of these areas below.

2.1 Emergency Navigation and Evacuation System

The problem of wireless sensor network (WSN) assisted navigation is probably firstly looked at by Li et al. [14], in which they develop distributed algorithms for self-organizing sensor networks that respond to guiding moving objects to a target location through a region while avoiding obstacles or dangerous areas. They use the sensor network to model the danger levels sensed across its field, which has the ability to react to the environment and adapt to changes. A distributed repository of information protocol is proposed in this paper and it can distributedly compute the safest path to direct the target (human or robot, equipped with a sensor that can talk to the network) to the goal with a safe distance to the danger areas. The protocol is validated via experiments on a 50-node sensor testbed under different topologies. In [15], Tseng et al. propose a distributed navigation algorithm based on the temporally ordered routing algorithm (TORA) for mobile ad hoc networks to guide people to a exit while helping them avoid hazardous areas in a building with a 2D layout during an emergency. The algorithm can quickly separate hazardous areas from safe areas and help sensors establish safe escape paths when the sensors detect emergency events. Each sensor is assigned an altitude based on its distance to the nearest exit node and the distance depends on the physical length and also the hazard intensity along with this path, i.e. nodes closer to exits have lower altitude and nodes near a hazard have high altitude. The approach offers users to be guided by minimizing the distance (i.e. number of hops) they have to spend in the hazard when multiple exits are available. Another advantage of this algorithm over the one proposed by Li et al. is that it has faster convergence of the navigation directions and lower communication overhead [9]. In [16] the work in [15] is extended to 3D indoor environment. Sensor nodes are deployed floor by floor and are connected together by those at stairs. In such applications, quick response is more critical and this paper propose the tree reconstruction protocol that reduces the occurrence of temporary cycles and shortens the convergence time. However, as in the case of Li et al., the authors only consider static hazards, and they focus on the calculation of paths and ignore the expected spread of the hazard inside the building [9].

One of the earlier work that studies both spreading hazards and evacuation dynamics is by Barnes et al. [17], who presents a distributed algorithm to find the safest paths for evacuees taking into account predications of the relative movement of hazards, such as fires and smoke, and evacuees. The paper presents a evacuation assistant application for wireless sensor networks, which aims to provide immediate and safe navigation routes in the event of a hazard through distributed route finding. By using the model, the spread of the hazard is predicted before the computation of the safe paths out of the building, and each node maintains and compute the time at which its location will be hazardous. The navigation algorithm computes the safety of a path to an exit node in terms of the margin of time an evacuee starting at the beginning of the path reaching each node before the hazard reaching those nodes. Although safe escape routes with considering the spread of a hazard can be found by using this system, a predetermined model of the movement of hazards and the movement of people in the building is required. The two weighted graphs over the network, called the “hazard graph” and the “navigation graph”, describe the spread of hazardous conditions and the speed of evacuation of persons, and in order to facilitate the operation of navigation algorithm, the data from both two graphs will be distributed throughout the network. However, since a single navigation graph is used, it is impossible to represent people with different moving speeds, different spreading paths, and different speeds of a hazard. Some of shortcomings of the system are addressed in [18], where the authors extends the concept safety introduced by Barnes et. al. [17] and a dynamic graph for emergency evacuation is introduced. In the dynamic graphs, the weight of an arc represents the predicted travel time along the arc and the arc weights vary in time depending on the presence of fire hazard. The drawback of the approach [18] is that the hazard information from the deployed sensors is centrally used to generate the dynamic graph and compute the evacuation paths, resulting in fault intolerant, and huge data transmission and computational requirements for large scale systems. Another work that uses a centralized approach for evaluation paths is presented in [19]. The authors proposes a indoor navigation system by using mobile terminals such as a cell phone or PDA, which provides an constant supervision of the situation of a living space and the detection of residents’ respective position. The notice for emergency evacuation is communicated using a cellular telephone internet connection service and the management of data transmission is realized by an integration service centre. People are assumed to be equipped with smart phones and beacon receivers that are designed to estimate the position of a user in indoor. The beacon receiver provides the location information via Bluetooth to the mobile phone handled by evacuees, which in turn sends it to the building service centre via Internet. The service centre also monitors building environment conditions, which provides real-time information about the hazard. Evacuation paths calculated by service centre are sent back to the mobile phones, which display the evacuation directions to the users. The system proposed by the authors is quite realistic, but suffers from high installation and management costs due to many communication technologies and devices, such as wireless sensor network, Bluetooth, 3G or WiFi infrastructure, and beacon devices. Also, the use of these devices leads to a high electricity consumption and carbon dioxide emissions.

A distributed building evacuation support system is proposed in [20] and particularly in [21]. In [20], the authors present a distributed decision support system for building evacuation, which aims at providing directions to evacuees during an emergency situation. The system is able to function in real-time, adapt to the changes of the environment and provide reliable and safe paths to evacuees towards the best available exit, which is evaluated by the use of a multi-agent Building Evacuation Simulator [21] that models the evacuation procedure. The proposed system is composed of a network of decision nodes and sensor nodes, and each decision node executes the distributed shortest path algorithm inspired by [7, 22], in a distributed manner. The existence of a network of sensor nodes provides the Decision Nodes with real-time information about the conditions inside the building and the Decision Nodes provides dynamic directions to people in its vicinity to evacuation exits. Simulations of the system take into account physical congestion, the number of evacuated civilians and average remaining evacuee health in various emergency scenarios. Results illustrate that the existence of the decision support system minimizes the evacuation time and injuries. The Distributed Building Evacuation Simulator in [21] addresses the unique needs for the simulation of emergency response scenarios. This simulator facilitates the modelling of diverse and autonomous agents, and it operates in a distributed fashion to reduce the simulation time required for large-scale systems. The simulation tool is evaluated in both centralised and distributed execution and the results show the significant reduction in execution time that is achieved for different degrees of distribution of the simulator on multiple serves. Previous work [23] discusses the use of analytical performance models, in which they consider the complex programming problem and enhance previous methodologies, queueing network models, by adding state-dependent queueing models to capture the non-linear effects of increased occupant traffic flow along emergency evacuation routes. More recently analytical model is introduced in [24] and it presents the use of analytical techniques based on not only queueing network models but also graph theory to improve the planning of building evacuation. The graph theory techniques can offer insight into the critical areas, and identify and then selectively reinforces sensors that are particularly important. Work on the design and optimisation of emergency management schemes based on discrete event simulation is challenged by the substantial amount of programming or reprogramming when existing simulation tools are applied to a new problem, the scalability, and the computing time needed to obtain useful performance estimates in realistic environments. Thus the authors suggest the use of graph and analytical model, which can provide fast estimates for the location of points of congestion, for estimating the sensitivity of outcomes to the presence of hazards, and for routing optimisation.

Routing emergency evacuees with Cognitive Packet Network (CPN) that alleviates slow convergence time for large and fast-changing networks is proposed in [4]. The authors consider a routing algorithm that is able to self-monitor and constantly make swift adjustments instead of requiring a full graph search each time the environment changes. CPN [7] lets each network node send a small flow of "Smart Packets" (SP) to monitor network conditions and discover new route to the exits using reinforcement learning [22,25] with a recurrent Random Neural Network (RNN) [26]. Thus, CPN algorithm can optimize any measurable Quality of Service (QoS) metric, not only shortest path mentioned in [21] but other metrics such as safety, congestion. Experiments take into account the comparison of three scenarios, "autonomous" (i.e. evacuees do not receive any assistance), Dijkstra's Shortest Path algorithm, and CPN algorithm separately. The results show that CPN reaches the performance level of the Dijkstra's Shortest Path algorithm in lower densities situations. As the density of evacuees increases, the Dijkstra's SP algorithm tends to simultaneously send all evacuees through one single optimal path resulting in the highest levels of congestion, while CPN has tendency to disperse users which in turn slightly reduces the congestion. However, results also reveals that the assistance systems progressively lose their advantage and become detrimental to the evacuees in high-density situations due to congestion forming in the building. What's more, in [8], the authors consider the problem of providing emergency support when existing communication infrastructure is destroyed or fail. They propose the use of opportunistic communications (oppcomms) among mobile devices carried by civilians for the dissemination of emergency information [9], and they describe an autonomous emergency support system (ESS) based on oppcomms to support evacuation of civilians in urban emergencies. With oppcomms, devices can exchange message packets at a close range of a few to some tens of meters with limited or no infrastructure and messages are carried over multiple hops in a "store-carry-forward" manner [9] by exploiting human mobility. The authors compare three different scenarios in system evaluation: (i) ESS without alarm (ESS^-), (ii) ESS with alarm (ESS^+), and (iii) ideal evacuation. Results show that the performance of the ESS is dependent on population density and the system perform much better on the metric of percentage of evacuated civilians for higher population densities. Also, ESS guides evacuees via paths that reduce exposure to the hazard and increase average evacuee remaining health even for low densities. ESS^+ performs as well as the ideal evacuation method, since it alerts civilians of the hazard as soon as possible for successful evacuation. The results on evacuation time illustrates that ESS^- takes longer evacuation times, and ESS^+ and ideal evacuation have similar performance on relative short evacuation times. With increasing civilian density, average evacuation times increase in ESS^+ and ideal evacuation scenarios while decreases in ESS^-, since in ESS^- civilians start the evacuation process at different times, which eases congestion during evacuation. Also, results indicate that the proposed ESS is well suited for deployment in urban environments with dense populations; ESS requires large communication ranges for acceptable performance in very sparse populations.

2.2 Rescue System for Emergency Management

The above work focuses on how to direct evacuees to evacuation exits effectively and reliably under different scenarios and topologies, however, people might find no way out due to fire or other obstacles, or they are injured and cannot move in real emergency environments, and then they have to wait for rescuers to help them evacuate to exits [11]. Thus, the search and rescue systems [2] are also necessary in emergency management, which aims to different objectives, tracking hazards and the movement of civilians, toward the events that are taking place [1], and aiding the victims. In [27], the authors address robot deployment to connect as many civilians as possible with a static base station. [28] proposes the use of robot-sensor network system to track injured civilians autonomously without relying on localization technologies. [1] models the movement of robots from colder sensors toward the hotter sensors so that the robots are closer to victims and can provide help to them. However, in this report we consider the application of emergency personnel, rescuers (human entity), to emergency situations when evacuees are trapped and immobilitied. This application is based on such distributed simulator and analytical performance model mentioned above which facilitate the modelling of rescuers' work and interaction between evacuees and rescuers in building evacuation situations. In rescue systems, identifying the locations of the possible victims is usually first step before executing rescue and an overestimate of victims may lead to resource wastage and more serious casualties [1]. Thus, the problem of how to efficiently assign rescuers to injured people is presented. Gelenbe et al. [12] purposes that Random Neural Network (RNN) with synchronized interactions can provide a fast approximate decision for a resource allocation that arises in emergency management, where they consider the problem whose exact solution would require an impractically large computation time. The approach taken is to train the RNN with the supervised learning algorithm using numbers instances of the optimization problem, with exact solutions that are obtained offline, and the trained RNN is then tested with randomly generated instances of the optimization problem. The output variables are represented by the use of two neurons: a "positive" neuron and a "negative" neuron. If resource i is allocated to incident j, then the excitation level of the corresponding positive output neuron is high, while the excitation level of the negative output neuron is low. If resource i is not allocated to incident j, then the excitation levels of the corresponding neurons are antisymmetric to the first case. This model is validated via hundreds of experiments and it shows that the RNN with triggered interactions yields results that are quiet close to the optimum value, as many casualties as possible being evacuated in emergency with short response times. We apply the idea of distributed RNN algorithm to the problem of emergency personnel allocation to maximise the number of injured collected and avoid the waste of emergency personnel. However, the individuals rushing for the exits along with safe paths calculated by the same algorithm may lead to congestion at bottleneck nodes, such as stair nodes and exit nodes, and even worse the rescuers who towards the hazard and aid victims will aggravate the congestion problem. [29] proposes a approach taking both pedestrian congestion and rescue force flexibility into account by providing firemen rescue commands to eliminate key dangerous areas. The emergency environment is represented as a directed graph and the human's movements are modelled as network flows on the graph. By calculating the maximum flow and minimum cut on the graph, the system can direct fireman to critical hazard locations to eliminate the danger which would alleviate congestion the most. Thus, the system in turn send the response information to civilians to disperse them to repaired paths. However, in fact, evacuees normally do not have the luxury of the presence of emergency personnel when the emergency starts, thus, an evacuation support system should be able to navigate people safely and quickly without relying on emergency personnel behaviours [9].

Chapter 3

Emergency Rescue Navigation for Emergency Management

3.1 Rescue System Model

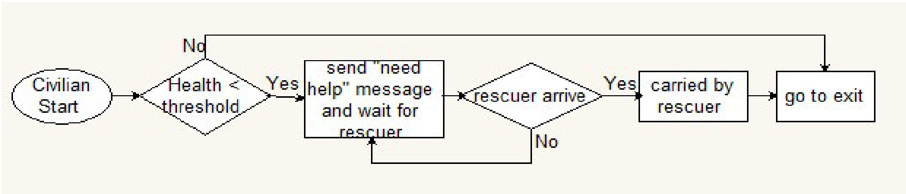
We begin by describing the design details and the assumptions we made of this system. A number of pervious works attempt to provide evacuees with the shortest and safest path towards the evacuation exits in dynamic hazard emergencies. However, such a path far away being enough to guarantee successful evacuation for all people [29]. Because when an emergency happens, all people in dangerous regions are trying to get out while some of the escape passageways have been obstructed by fire, smokes, obstacle, and so forth, so that people have no way to move out and have to wait for help. Also, it is very likely that the evacuees who are near hazardous areas are seriously injured and even immobilitied in dynamic disaster environments. Thus, the rescue system model is designed to provide support to those trapped civilians and lead or take them out of dangers.

The rescue system model is established based on Distributed Building Evacuation Simulator (DBES) [21], where all simulated entities are modelled as agents, including evacuees as well as rescuers. These entities are able to communicate with each other, which is realized by using a well-defined communication ontology, and all interactions follow the FIPA standards [30]. Civilians try to get out the hazard along with a safe path to avoid injure, while rescuers aim to get into hazardous areas to save people and then carry them to safe sites. Here, the assumptions we made are presented firstly. The first assumption is that the moving speed of rescuers is faster than that of civilians, and this is due to the fact that rescuers are emergency personnel who are normally strong. Another assumption is that the initial health status of civilians and rescuers is 100, and the health is affected by the hazard, i.e. their health level decrease gradually if they expose to hazardous areas. In real emergencies, the rescuers will wear protective clothes so that they can expose to the hazard relative long time than civilians. Thus, we assume that the health level of rescuers decreases slower than civilians. Additionally, if the remaining health of civilians is less than a threshold, we assume that they cannot move, and will stop and wait for rescuers, so that their health level decreases slower than moving. Because civilians can protect themselves from the danger for a while by hiding themselves in a relative safe place and using clothes. Moreover, the civilians are assumed to be safe during the stage that rescuers are carrying them to the exits. We also assume that each rescuer will be allocated one trapped civilian each time but it allows for the possibility that one more trapped civilian may be assigned to a rescuer if the location of the trapped civilian is near the way selected by the rescuer towards the exits. In other words, the capacity of each rescuer is one and the maximum capacity of each rescuer is two, but the maximum capacity can be reached only if the injured civilian is the neighbour of the nodes which the rescuer gets through to evacuation exits. Furthermore, the rescuers can continue work after finishing one task if their health level is good enough to do so. The last assumption is to claim the departure conditions of rescuers. The rescuers will leave the building if there is no civilian inside the emergencies and/or if their health level is not good enough to rescue people. This assumption is used to keep an eye on rescuers health status in order to avoid rescuer sacrifice.

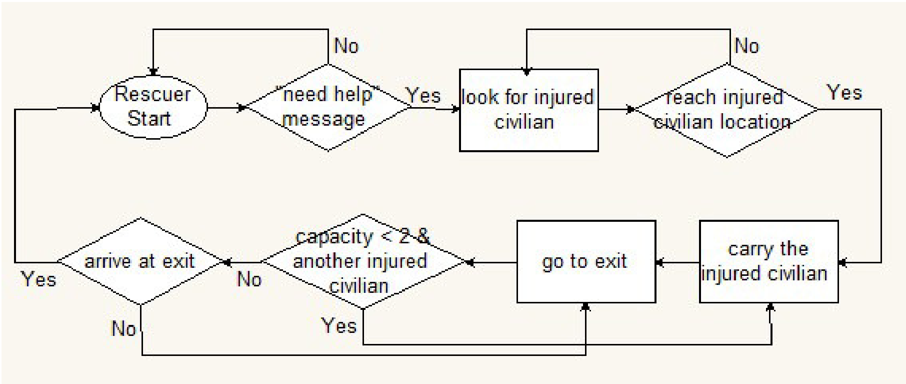
The navigation algorithm used by civilians and rescuers is Dijkstra's Algorithm which can compute the shortest path from a given node to another given node. Here, the shortest path is regarded as the most quickly path since moving speeds of both civilians and rescuers are fixed. For civilians, their goal is to evacuate from the emergencies quickly along with a safe road which is far away from the hazardous areas, thus, the length between two nodes should be computed based on not only its physical length but also the hazard intensity on the link. Therefore, the path selected by civilians can navigate them to evacuation exits as quick as possible, and also make them away from disaster regions to keep them safe as far as possible. For rescuers, the navigation algorithm is applied to two stages of rescue process, searching for injured civilians and carry injured to the exits. During the first stage, the rescuers try to arrive at emergency sites quickly to assist the injured civilians in real time, and we assume that they cannot be hurt when they get through the hazard themselves since they wear protective clothes. Therefore, the navigation algorithm for this stage does not need to consider the hazard intensity on each link and compute the shortest path only based on the physical length between nodes. However, the rescuers take injured people to the exits in the second stage, so that the rescuers should not only go to the exits as quick as possible but protect people from further injuring. Thus, the navigating algorithm used in this stage is similar with the one used for civilians evacuation since the algorithm should also consider the physical length of a path as well as the hazard intensity on it, however, the difference is that the dominated metric of the algorithm is the hazard intensity because we expect the rescued civilians to be safe. The mathematical model of the navigation algorithms mentioned in this part is presented in detail in Section 3.2.

3.2 Description of Rescue Process

The procedure of rescue task is presented in this section. An overview of the rescue operation is illustrated in Figure 3.1, and it can be noticed that the status and behaviour of civilians have effects on rescuers and vice versa. The rescuers start their work when they receive "need help" message from the sensor network, i.e. if there are people who are injured and unable to move in emergencies, the rescuers will commence rescue immediately. The civilians, when they are injured and need to be rescued, we assume that they will stop moving and wait for rescuer. Because it is not easy and efficient to search for a person if he or she is moving. Additionally, the civilians will follow the rescuer once they have been saved, and this is due to the fact that the injured civilians will be taken or led by



(a) Civilian Goal



(b) Rescuer Task

Figure 3.1: Flowchart representing the operation of the rescuing procedure of (a) civilians and (b) rescuers

rescuers toward evacuation exits. Thus, the basic information of trapped evacuees, such as their identity IDs and locations, needs to be sent to the rescuers, so that the rescuers can identify the injured civilians and try to find them. On the other hand, the civilians should know where the rescuers are and if the rescuers have arrived at. Therefore, the information exchange between civilians and rescuers is important in emergency situations. We suppose that there are mobile computing devices carried by both civilians and rescuers, connected to other computer networks via a wireless infrastructure network, providing emergency communications and distributed management. In order to realise it, the corresponding initiator and responder is added into civilians' and rescuers' behaviour, the goal of initiator is to trigger a action to sense the changing of the environment and if the initiator receives a message, then the message will be sent to the responder as a request, after processing, the responder will generate a response and the initiator will handle this information and then take the next action. In this way, the initiators and responders can provide real-time emergency information to both rescuers and civilians. The three phases of rescuing process are illustrated below:

3.2.1 Phase I Triggering rescuers

When the fire starts, civilians evacuate towards the exits by executing Dijkstra's shortest path algorithm with considering the hazard intensity. As mentioned in Section 3.1, the length between two nodes is computed based on its physical length as well as the hazard intensity on the link. The previous work proposed the idea of "effective length" $L(i, j)$, which is defined of a link as [20]

$$L(i, j) = l(i, j) \cdot H(i, j). \tag{3.1}$$

Equation 3.1 shows that the value of L depends both on the physical length of a link $l(i, j)$ and on the value of hazard along that link $H(i, j)$. When there is no hazard present, $H(i, j) = 1$, and the value of $H(i, j)$ will increase with observed hazard, thus, the higher the value of $L(i, j)$ on a link, the more hazardous it is for a civilian to move along it. Therefore, for each civilian, the algorithm ranks all paths from the location of the civilian to the exits based on the effective length between nodes along with each path, and then pick out the shortest one as the evacuation path for this civilian. Also, their health will be affected as they will come in contact with hazardous path sections, and when the value of health is below than a threshold, the civilian is injured, and they will stop moving and protect themselves from the hazard, so that their health will decrease slower than moving. Simultaneously, they send "need help" message to rescuers to trigger a rescuer starting rescue.

3.2.2 Phase II Identifying locations of injured civilians

Rescuer obtains the information of the injured civilian from the received message and goes to the corresponding location by executing shortest path algorithm without considering hazard intensity, since we expect that the rescuers can arrive at the location of injured civilian with the shortest time, and rescuers will carry protective equipment that can withstand exposure to the hazard up to a certain degree. In this case the hazard intensity $H(i, j)$ is set to 1 and the "effective length" $L(i, j) \equiv l(i, j)$, i.e. the "effective length" is equivalent to the physical length of the link. During this process, the single duty of rescuers is "decide-go", for example, once the rescuer decide to save civilian A, he will never change his way to save others, since it is not reality to decide if civilian A has died and if he always changes his direction to save someone near him, he maybe cannot save anybody. When the rescuer reaches the location of injured civilian, this civilian will be saved and assumed to be safe.

3.2.3 Phase III Carrying injured civilians to exit

At this phase, rescuers carry the injured civilian to one of the evacuation exits by executing shortest path algorithm with considering hazard intensity, since the civilian will be injured seriously if exposing to the hazard and we expect to navigate the injured people as far away from hazardous regions as possible, thus, rescuers will not only go to exit as soon as possible but also avoid hazardous path to maintain the health level of the injured civilians. In this case, the dominated metric of Dijkstra's Algorithm is the hazard intensity, i.e. a link with the lowest hazard intensity will be treated as the safest path and selected to be the next way towards exit, and a link with the shortest physical length will be considered if two or more links have the same hazard intensity. We have assumed that the maximum capacity of a rescuer is two, which means that a rescuer can save at most two people at the same time, also, we expect that the rescuer can arrive at exit quickly so that the injured civilian can receive timely treatment and the rescuer can save others. Therefore, the rescuers can only save one more people who is the neighbour of the path towards the evacuation exit. In this case, the maximum capacity of the rescuers may be not reach if locations of injured civilians are far away from each other. When rescuers carrying the injured civilians reach the exits, they check injured civilian list and if it is not empty, they continue rescuing. Otherwise, they will wait for the task until every evacuee has left the building, then they will exit as well. As mentioned before, rescuers are not expected to sacrifice, thus, if their health level is lower than a threshold, we suppose that they cannot continue working and they leave the building immediately.

3.3 Task Assignment Algorithm

When a limited set of rescuers needs to be allocated simultaneously to a set of injured civilians, the problem of how to dispatch rescuers to save people maximally and efficiently needs to be considered, and this is a kind of task assignment problem. In this section, task assignment problems are introduced firstly, and then we present that random neural network (RNN) based algorithm can provide a fast and near-optimal solution to this kind of problem.

Assignment problems involve optimally matching the elements of two or more sets [31], and in this report only two sets will be considered, which refers to as "tasks" and "resources". In this emergency rescue system, "tasks" is injured civilians who need the help of evacuation and "resources" is rescuers that they can aid injured civilians and carry them to building exits. The assignment problem includes assigning each task to a different resource, with each resource being allocated to at most one task (a one-to-one assignment) or to multiple tasks (a one-to-many assignment) [31]. This rescue model assumes that one rescuer will be allocated to one trapped civilian, but it allows for the possibility that more than one trapped civilian may be assigned to one rescuer if the capacity of rescuers is more than one person (i.e. rescuers are able to carry more than one people toward evacuation exits at the same time). There is a cost associated with each possible assignment of a resource to a task and the objective of task assignment problem is to minimise overall cost of resource allocations.

The problem of finding one-to-one matching between n tasks and m resources will be described firstly. The mathematical model for this problem may be given as [31]:

$$Minimize \qquad \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \tag{3.2}$$

Subject to: (1) $\sum_{i=1}^{n} x_{ij} = 1 \quad j = 1, ..., n;$ (2) $\sum_{j=1}^{n} x_{ij} = 1 \quad i = 1, ..., n,$

$$(3) x_{ij} = 0 \;\; or \;\; 1,$$

where x_{ij} is a binary decision variable with $x_{ij} = 1$ if resource i is allocated to task j and $x_{ij} = 0$ if not, and c_{ij} is the cost of assignment of resource i to task j. The first constraint ensures that every task is assigned to only one resource and the second one ensures that every resource is allocated to a task. The one-to-one matching problem is mathematically identical to the *weighted bipartite matching* problem from graph theory [31], so that the results from *weighted bipartite matching* problem can be used in constructing efficient solution procedures for task assignment. Another type of problem is that assigning more than one task to the same resource is allowed or required, thus, a model with multiple tasks per resource will be considered. This one-to-many assignment model has been discussed in [31], and the authors call it generalized assignment problem (GPA), where it is possible to allocate the same resource to two or more tasks with the limitation of capacity of the resource. This model will be described following [31] and we assume that there are n tasks and m resources needing to be matched:

$$Minimize \qquad \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \tag{3.3}$$

Subject to: (1) $\sum_{i=1}^{m} x_{ij} = 1 \quad j = 1, ..., n;$ (2) $\sum_{j=1}^{n} a_{ij}x_{ij} \leq b_i \quad i = 1, ..., m,$

$(3) x_{ij} = 0 \;\; or \;\; 1,$

where $x_{ij} = 1$ if resource i is allocated to task j and $x_{ij} = 0$ if not, c_{ij} is the cost of assignment of resource i to task j, a_{ij} is the amount of resource i's capacity used if this resource is allocated to task j, and b_i is the total available capacity of resource i. The first constraint makes sure that every task is assigned to only one resource and the second one makes sure that a set of task assigned to a resource do not exceed its available capacity. Note that although there m constraints in the second one, one for each resource, i.e. every resource's capacity for doing these tasks is being considered. For the solution of the one-to-many assignment problem, Bokhari [32] shows how, in many problems with special, although normally occurring, structures of solutions can be relatively easily determined using methods from graph theory.

3.3.1 Random neural network

In emergency situations, evacuees cannot expose to the spreading hazard, such as fire and smokes, long time, and this is due to the fact that longer waiting time in dynamic hazard environments means less chance to survive. Therefore the search and rescue systems require a mechanism that can solve the optimization problem of allocating a set of tasks to a limited set of resource simultaneously very quickly and quiet accurately. The random neural network (RNN) [12], a probabilistic model with a well-developed mathematical theory, inspired by the apparently stochastic spiking behaviour of certain natural neuronal systems, can satisfy such requirements and provide very effective resource allocation solution. In [12], the authors discuss an extension of the random neural network model (RNN), which make it possible that cells synchronously act together on other cells, triggering successive firing instants in other cells, with incorporating the usual excitatory and inhibitory cells' interactions. The RNN with synchronized interactions can offer a fast approximate decision for a resource allocation in emergency management, with limited computational resources and in real time. The approach taken for fast optimization in task assignment problems is to train the RNN with the learning algorithm derived in [12], with exact solutions that are obtained offline, and the trained RNN is then used to do resource allocations. The basic idea for the RNN training purposed in [12] are shown below firstly.

Table 3.1: List of RNN training model symbols

Notation	Definition
r_i	Firing rate for neuron i
$k_i(t)$	Internal state of neuron i
$\Lambda(i)$	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

The main symbols used for RNN training model are summarised in Table 3.1 and an application of one earlier result (Theorem 1) proposed in [33] is being used for RNN training.

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

$$\lambda^-(i) = \lambda(i) + \sum_{j=1}^{N} r_j q_j \left[p^-(j,i) + \sum_{m=1}^{N} Q(j,i,m) \right] \tag{3.4}$$

$$\lambda^+(i) = \sum_{j=1}^{N} r_j q_j p^+(j,i) + \sum_{j=1}^{N} \sum_{m=1}^{N} q_j q_m r_j Q(j,m,i) + \Lambda(i), \tag{3.5}$$

where

$$q_i = \lambda^+(i)/(r_i + \lambda^-(i)). \tag{3.6}$$

If a unique nonnegative solution $\{\lambda^-(i), \lambda^+(i)\}$ exists for the nonlinear system of equations, 3.4 to 3.6, such that $q_i < 1 \ \forall i$ then

$$\pi(\underline{k}) = \prod_{i=1}^{N} (1 - q_i) q_i^{k_i}. \tag{3.7}$$

There will be a solution if the network is in steady state, and the stability condition is $q_i < 1$ that guarantees that the excitation level of each neuron remains finite with probability one. According to the definition of $p^+(i,j)$, $p^-(i,j)$, $d(i)$, and $Q(i,j,m)$ in Table 3.1, we have a relationship between four different types of probabilities.

$$d(i) = 1 - \sum_{j=1}^{N} \left[p^+(i,j) + p^-(i,j) + \sum_{m=1}^{N} Q(i,j,m) \right]. \tag{3.8}$$

We now use notation “weight” similar to that used in [26] to represent the rates at which the neuron interacts, and the weights are defined as a simple product form of firing rate and corresponding probability.

$$w^+(i,j) = r_i p^+(i,j), \tag{3.9}$$

$$w^-(i,j) = r_i p^-(i,j), \tag{3.10}$$

$$w(i,j,l) = r_i Q(i,j,l). \tag{3.11}$$

Combine equation 3.8, 3.9, 3.10, and 3.11, we can rewrite equation 3.8

$$
\begin{aligned}
r_i(1-d(i)) &= r_i \sum_{j=1}^{N} \left[p^+(i,j) + p^-(i,j) + \sum_{m=1}^{N} Q(i,j,m) \right] \\
r_i(1-d(i)) &= \sum_{j=1}^{N} \left[w^+(i,j) + w^-(i,j) + \sum_{m=1}^{N} w(i,j,m) \right] \\
r_i &= \frac{\sum_{j=1}^{N} \left[w^+(i,j) + w^-(i,j) + \sum_{m=1}^{N} w(i,j,m) \right]}{1-d(i)}. \qquad (3.12)
\end{aligned}
$$

The denominator of q_i in equation 3.6 then can be rewrited combining with 3.4, 3.10, and 3.11

$$
\begin{aligned}
D(i) &= r_i + \lambda^-(i) \\
D(i) &= r_i + \lambda(i) + \sum_{j=1}^{N} r_j q_j \left[p^-(j,i) + \sum_{m=1}^{N} Q(j,i,m) \right] \\
D(i) &= r_i + \lambda(i) + \sum_{j=1}^{N} q_j \left[r_j p^-(j,i) + \sum_{m=1}^{N} r_j Q(j,i,m) \right] \\
D(i) &= r_i + \sum_{j=1}^{N} q_j \left[w^-(j,i) + \sum_{m=1}^{N} w(j,i,m) \right] + \lambda(i), \qquad (3.13)
\end{aligned}
$$

similarity its numerator becomes if combining with 3.5, 3.9, and 3.11

$$
\begin{aligned}
N(i) &= \lambda^+(i) \\
N(i) &= \sum_{j=1}^{N} r_j q_j p^+(j,i) + \sum_{j=1}^{N} \sum_{m=1}^{N} q_j q_m r_j Q(j,m,i) + \Lambda(i) \\
N(i) &= \sum_{j=1}^{N} q_j w^+(j,i) + \sum_{j=1}^{N} \sum_{m=1}^{N} q_j q_m w(j,m,i) + \Lambda(i), \qquad (3.14)
\end{aligned}
$$

$$
\textit{so that} \quad q_i = N(i)/D(i). \qquad (3.15)
$$

The results now can be used to design an efficient learning algorithm by updating weights of all neurons and selecting the neuron who has the largest q.

In building evacuation systems, N_L incidents occur simultaneously at different locations with I_j civilians injured at incident j. N_U is the number of rescue personnel who are spatially distributed at evacuation exits before the time of incidents, and each rescuer is able to collect $c_i > 0$ injured people and the response time of managing incident j by rescuer i is $T_{ij} > 0$. If the total capacity of rescuers is sufficient to collect all trapped civilians, then the objective of this problems is not only to rescue victims as many as possible but also to minimize the average response time of rescuers. We assume that only one rescuer can be allocated to one incident and that the initial locations of rescuers and the locations of incidents are known and fixed. The mathematical model is given as [12]:

$$Minimize \qquad \sum_{i=1}^{N_U} \sum_{j=1}^{N_L} T_{ij} x_{ij} \tag{3.16}$$

Subject to: $\quad (1) \sum_{j=1}^{N_L} x_{ij} = 1 \quad i = 1, ..., N_U; \qquad (2) \sum_{i=1}^{N_U} c_i x_{ij} \geq I_j \quad j = 1, ..., N_L,$

$$(3) x_{ij} = 0 \ \ or \ \ 1,$$

where $x_{ij} = 1$ if rescuer i is allocated to incident j and $x_{ij} = 0$ if not, c_{ij} is the cost of assignment of rescuer i to incident j, T_{ij} is the response time of assigning incident j to rescuer i, and I_j is the number of injured civilians at incident j. The first constraint indicates that a rescuers must be allocated to exactly one incident to avoid resource wastage, and the second constraint expresses the fact that the available capacity of rescuers allocated to an incident should be at least equal to the number of victims there. The input variables are I_j, since rescue actions will be triggered if there are people injured and immobilitied, and I_j will be represented by the parameters $\Lambda(j)$ as the input neurons in RNN due to the fact that $I_j > 0 \ \forall j$. While the outputs are x_{ij} that are represented by the use of two neurons, a "positive" neuron referring to excitation and a "negative" neuron referring to inhibition. It indicates that (a) if $x_{ij} = 1$, then the excitation level of the positive output neuron is high, while the excitation level of the negative output neuron is low; (b) if $x_{ij} = 0$, then the excitation level of the corresponding neurons are antisymmetric to (a). In the learning procedure, the wight of each neuron is updated by using RNN training model mentioned above and the whole procedure is repeated until convergence. At that moment, this trained model can provide the fast optimization in resource allocation problems.

3.3.2 Description of task assignment algorithm

Although the rescuers can identify the location of injured civilians relying on the received messages from the sensor network, it is difficult to make a precise prediction on the changing of inside environment in dynamic hazard emergencies, such as hazard intensity, health status of the injured civilians, and so forth, so it is inappropriate to assume that the rescuers can always success in executing a mission. Thus, it is uncertain if a rescuer will reach the location of the injured civilian and carry them to one of evacuation exits, i.e. in an uncertain environment a rescuer may fail in completing a task. Moreover, as mentioned in section 3.3.1, the resource allocation in emergency situations requires fast and near-optimal algorithms to provide advice in real-time [1]. Therefore, the requirement of task assignment algorithm in emergency situations is to provide the near-optimal advice in real-time in uncertain environment. In [34], the related resource allocation problems are formulated as a non-linear optimization problem, which is then solved approximately in real-time based on the trained Random Neural Network mentioned in section 3.3.1.

Consider a group of rescuers R waiting at building exits and some victims V in hazardous areas, and assume that the initial locations of both rescuers and injured civilians are known. The rescuers are able to potentially rescue any of injured civilians, but they rescue injured civilians with a probabilistic outcome of success. There is a cost $C(r, v)$ associated with each possible assignment of a rescuer r to a victim v, and if a victim is not allocated a rescuer there is a penalty denoted $K(v)$ for not executing rescue for the victim v. It is also possible that a rescue action may fail despite the fact that a rescuer has been assigned to the victim, and this will be represented by the probability $q(r, v)$ that a rescuer r is unable to rescue victim v, or in other word the rescuer r is not perfectly effective or suited to deal with the needs of victim v. Here, $0 \leq q(r, v) \geq 1$. Additionally, in order to reduce the probability of rescue failure more than one rescuers may be allocated to the same victim. Thus, any assignment of rescuers to victims will result in an expected overall cost and the objective of the algorithm is to minimize the overall cost in order to maximize the number of injured civilians collected and minimize the response time. We represent the decision of assigning the rescuer r to the 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. Hence,

$$p(r, v) \in \{0, 1\}, \quad and \quad \sum_{v \in V} p(r, v) = \pi(r) \leq 1, \quad r \in R, v \in V, \tag{3.17}$$

where $\pi(r) \leq 1$ represents that (a) we may decide not to assign the rescuer r to some victims if the assignment will increase the expected overall cost, and (b) a rescuer cannot be re-assigned some other victims if the rescuer has been allocated to some victims. For (b), we are considering cases where the resources (i.e. rescuers) are expendable and cannot be re-used, which will be realized by removing rescuers who have task from the resource list.

The number of victims and rescuers are denoted as $|V|$ and $|R|$, respectively. Since the cost of allocating rescuer r to victim v is $C(r, v)$, and the penalty of not rescuing the victim v is $K(v)$, the overall cost function that we try to minimize can be written as:

$$Minimize\ C = \sum_{v \in V} \sum_{r \in R} C(r, v) p(r, v) + \sum_{v \in V} K(v) \prod_{r \in R} \{1 - (1 - q(r, v)) p(r, v)\} \tag{3.18}$$

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

where the first term of equation 3.18 is the average cost of the rescuer allocation, and the second term is the accumulative average cost of not successfully executing rescue for each victims [34]. Equation 3.18 implicitly assumes that the decision to allocate one rescuer to a victim has no effect on the decision to allocate another rescuer to the same victim, in other word, each rescuer can be assigned without coordination among other rescuers. This realizes the distributed decision making function. Also, the product of the failure probabilities, $\{1 - (1 - q(r, v)) p(r, v)\}$, over all rescuers indicates that the allocation of different rescuers to the same victim v has a cumulative independent effect as to the overall success. This is due to the fact that if a rescuer is able to save the victim, cooperating with other rescuers will not increase the probability of success. The first constraint ensures that we can allocate one particular rescuer to at most one victim, thus, the total number of allocations can be less than $|R|$. 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 failure probability of equation 3.18 can be rewritten in exponential form,

$$
\begin{aligned}
F &= 1 - (1 - q(r, v))p(r, v) \\
F &= 1 - (1 - q(r, v))(p(r, v) = 0 \; or \; 1) \;\; (constraint(2)) \\
F &= 1, \qquad if \; p(r, v) = 0 \\
&= q(r, v), \; if \; p(r, v) = 1 \\
Thus, \; F &= q(r, v)^{p(r,v)}.
\end{aligned} \tag{3.19}
$$

Substitue equation 3.19 into equation 3.18, we obtain,

$$
Minimize \; C = \sum_{v \in V} \sum_{r \in R} C(r, v)p(r, v) + \sum_{v \in V} K(v) \prod_{r \in R} q(r, v)^{p(r,v)}, \tag{3.20}
$$

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

In order to find the minimum cost solution of equation 3.20, it is possible to enumerate all possible allocations of rescuers to injured civilians, and then select the optimal result. Because of the high computational cost, the RNN algorithm described in section 3.3.1 is used to solve equation 3.20, and this RNN parameters associated approach can provide a quick and accurate solution. The RNN model has been introduced in section 3.3.1, and here each possible assignment decision (r, v) is represented by a neuron $N(r, v)$ of a RNN, and the computational size of the approach is therefore $|R| \times |V|$. Firstly, a list of symbols used for the approach is summarized in Table 3.2. To specify the RNN which is used for the heuristic solution to this optimization problem, we specify excitatory and inhibitory signals' arrival rates of each of neurons $N(r, v)$, and the rate of excitation and inhibition signals between neurons below [34]:

Table 3.2: List of RNN parameter association approach symbols

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 rescuing victim *v* using 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

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

where $b(r, v)$ is the net expected reduction in the cost objective function (Equation 3.20) when rescuer *r* is allocated to victim *v*, since $K(v)(1 - q(r, v))$ is the expected reduction in the penalty of not rescuing victim *v* if this allocation has already made and $C(r, v)$ is the cost of allocating rescuer *r* to victim *v*. $b(r, v)$ also can be viewed as a characteristic of the pair *(r, v)*, where this allocation can minimize the overall cost of task assignments if $b(r, v) > 0$ (i.e. the expected reduction in the cost of saving victim *v* by triggering rescuer *r* is larger than the cost of allocating rescuer *r* to victim *v*, and then we think this allocation is effective), and this allocation is not the optimal one if $b(r, v) \leq 0$. To avoid resource wastage, we discourage the allocation of more than one rescuers to one victim, thus, we set the inhibitory weights:

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

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 allocation 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 one rescuer:

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

To keep matters as simple as possible, we choose not to reinforce or weaken any of the assignments than what is already done via the incoming excitatory signals [34], 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. Hence, we rewrite Equation 3.12,

$$\begin{aligned} r(r, v) &= \frac{\sum_{r', v'} [w^+(r, v; r', v') + w^-(r, v; r', v')]}{1 - d(r, v)} \\ r(r, v) &= \sum_{r', v'} w^-(r, v; r', v'). \end{aligned} \tag{3.21}$$

Equation 3.15 is then rewritten:

$$\begin{aligned} Q(r, v) = \Lambda(r, v) \quad &/ \quad [\lambda(r, v) + r(r, v) + \sum_{r', v'} Q(r', v') w^-(r', v'; r, v)] \\ Q(r, v) = \Lambda(r, v) \quad &/ \quad [\lambda(r, v) + r(r, v) \\ &+ \quad \sum_{r' \neq r} Q(r', v) w^-(r', v; r, v) + \sum_{v' \neq v} Q(r, v') w^-(r, v'; r, v)]. \end{aligned} \tag{3.22}$$

When there are civilians who are injured and cannot move, the sensor network will receive "need help" messages from injured civilians and this message also includes sources' ID, location and health status, and then the sensor network forwards the messages to rescuers. Next, the system of equation 3.22 is solved iteratively in the following manner to obtain the optimal assignments of rescuers to victims:

Algorithm Description:

1. Initilization: initialize (a) the set of rescuers remaining to be assigned R_{rem} to the set of rescuers R; (b) the solution set S where the assigned rescuer-victim pairs are stored to empty; (c) the penalty of injured civilians to $K_{cur}(v),\ \ v \in V$; (d) the fail probability $q(r, v)$ for each neuron, $r \in R, v \in V$; (e) the cost $C(r, v)$ for each neuron according to the distance between rescuer r and victim v, and the current hazard intensity around victim v, $r \in R, v \in V$; (f) $Q(r, v)$ for $r \in R, v \in V$ to zero: assume that all possible assignments have no effect on the cost of the objective function at

the beginning.

2. Compute RNN parameters $\Lambda(r,v)$, $\lambda(r,v)$, $w^-(r,v;r',v)$, $w^-(r,v;r,v')$, and $r(r,v)$, and construct the neural network for $r \in R_{rem}$ and $v \in V$.

3. Iteratively compute $Q(r,v)$ for $r \in R_{rem}$ and $v \in V$ based on the equation 3.22 until $Q(r,v)$ convergence for all neurons.

4. Select rescuer-victim pair (r^*, v^*) whose probability of being active is highest; if all $Q(r,v) = 0$ for $r \in R_{rem}$ and $v \in V$ then stop: there is no assignment that can reduce the cost of the objective function.

5. Update the solution set S by adding new rescuer-victim pair (r^*, v^*).

6. Update the remaining rescuers set R_{rem} by removing rescuer r^* from R_{rem}.

7. Reduce the penalty of the victim v^* by the expected reduction: $K_{cur}(v^*) = K_{cur}(v^*) - K_{cur}(v^*)(1 - q(r^*, v^*)) = K_{cur}(v^*)q(r^*, v^*)$.

8. Check the remaining rescuers set R_{rem}, and if R_{rem} is not empty then go to step (2), otherwise, stop: all rescuers has been allocated.

The description illustrates the main ideas of the algorithm and the Task Assignment Algorithm has shown below. Note that using this algorithm, the assignment of some rescuers r^* to a victim v^* always results in reducing the cost of the objective function; otherwise if $b(r,v) \leq 0$ then $Q(r,v) = \Lambda(r,v) = 0$ and the neuron is not selected [34]. One feature of the algorithm is that the rescuers can make decisions in a decentralized manner and arrive at a non-conflicting decision even though their actions are not coordinated. This is possible because the RNN algorithm is not stochastic, while the solution of the RNN signal-flow equations is unique [34]. Knowledge of the RNN parameters can be accomplished in an initial phase prior to decision making, in which each rescuer exchange with other rescuers any information associated with it [34].

Algorithm 1 Optimal task assignment for building evacuation systems

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:       $convergence = false$; // not convergence
20:       break;
21:    **end if**
22: **end for**
23: **if** $convergence$ **then**
24:    // go to line 28;
25: **else**
26:    // 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:       slect rescuer-victim pair (r^*, v^*);
33:    **end if**
34: **end for**
35: update $S = S \bigcup (r^*, v^*)$;
36: update $R_{rem} = R_{rem} \setminus r^*$;
37: update $K_{cur}(v^*) = K_{cur}(v^*) q(r^*, v^*)$;
38: **if** R_{rem} is not empty **then**
39:    go to line 2;
40: **else**
41:    stop;
42: **end if**

Chapter 4

Performance Evaluation using Simulation

4.1 The Impact of Rescuers in Evacuation System

The effectiveness of the rescuer for helping trapped civilians is evaluated by using the Distributed Building Evacuation Simulator (DBES), a Discrete-Event Simulator (DES). We assume that this is a fire-related emergency evacuations and the area to evacuate is a building, based on the three lower floors of Imperial College London's EEE building. A coarse graph representation of this building is showed in Figure 4.1. The performance of the system is evaluated by using the following metrics: (1) Percentage of survivors versus the evacuation time. This metric reflects the efficiency of the system in directing the civilians towards the exits, with respect to the speed of evacuation procedure. The faster the convergence of the curve, the shorter evacuation time, and the higher the curve, the more survivors; (2) Average percentage of evacuees that have exited the building. This metric reflects the effect on adding rescuers into the system, and the higher the bar, the more survivors; (3) Average remaining health of the evacuees. It denotes the efficiency of rescuers' work and a high value indicates that the injured civilians has succeed in being rescued timely, towards the exits along safe paths; (4) Percentage of evacuees who are saved by rescuers based on the total number of injured civilians. This is a straightforward metric

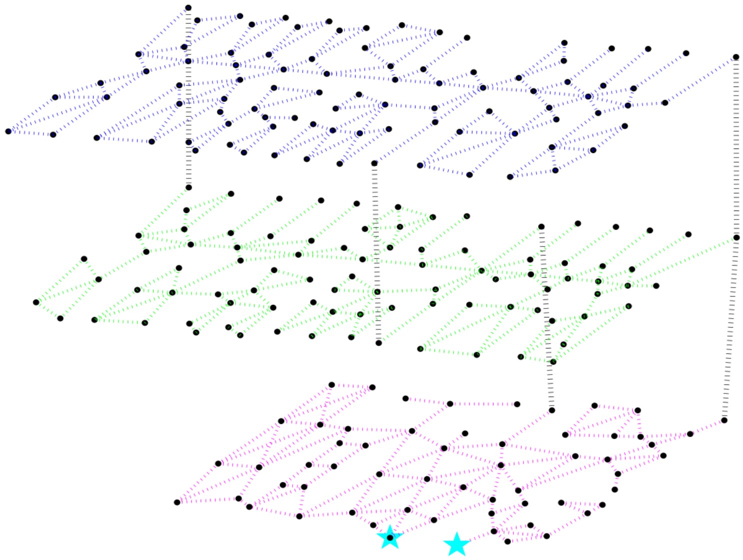
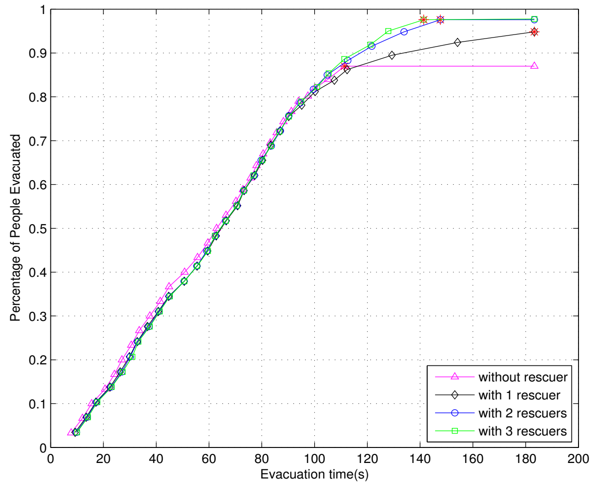


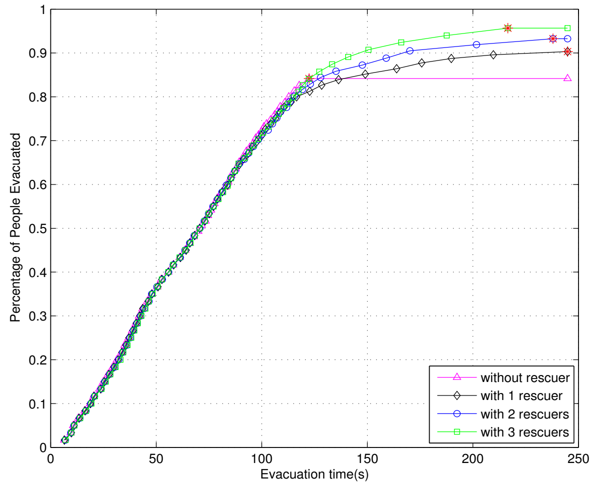
Figure 4.1: Graph representation of the building model. The two cyan stars on the ground floor mark the position of the building's exits.

which denotes the efficiency in rescuing injured people and carrying them to evacuation exits, and the higher the bar, the more trapped people are rescued. Our experiments feature four different scenarios for comparison, without rescuer, with one, two and three rescuers separately, and each scenario is executed 20 simulation runs. In order to obtain the general results, for each simulation run, we randomise the initial civilian locations and allocate the different types of civilians (adult, children and wheelchair) on nodes by setting different probabilities. This allows us to test the system under different building occupancy patterns. Moreover, we assume that when the rescuers finish one task, they can continue saving others if their heath status is able to do so. That is to say that a rescuer may get into the hazard twice or more times to save trapped people if they are able to continue rescuing.

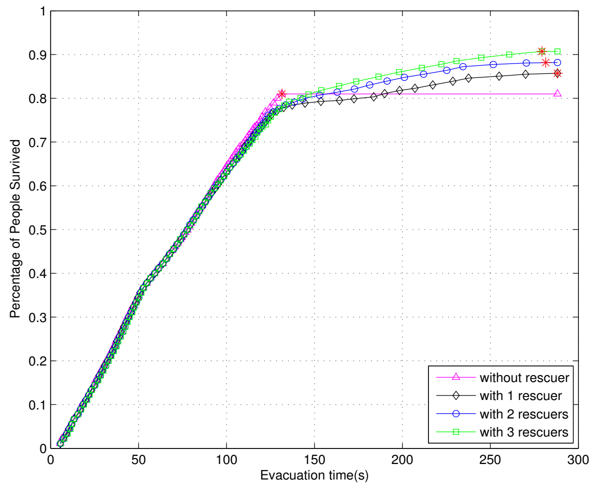
Figure 4.2 illustrates the percentage of survivors in the emergency versus the evacuation time, for different number of rescuers with varying population densities (from 10 to 30 per floor). Comparing with different cases (Fig. 4.2(a), (b) and (c)), as civilian density increases, we see that the evacuation takes longer time and the percentage of survivors becomes lower due to the congestion problem at bottleneck points in the building such as stairs [8]. For each case, we notice that in the scenarios of without rescuers, the speed



(a) 30Evacuees



(b) 60Evacuees



(c) 90Evacuees

Figure 4.2: Average percentage of survivors that have exited the building vs Evacuation Time, for different number of rescuers ('*' indicates convergence points).

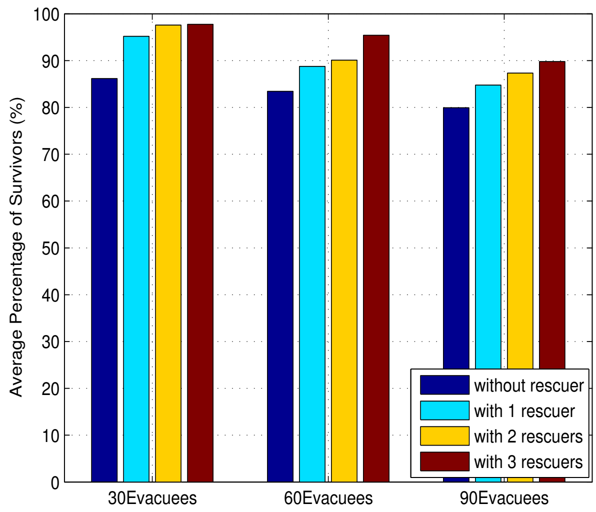


Figure 4.3: Average percentage of evacuees that have exited the building for each scenario.

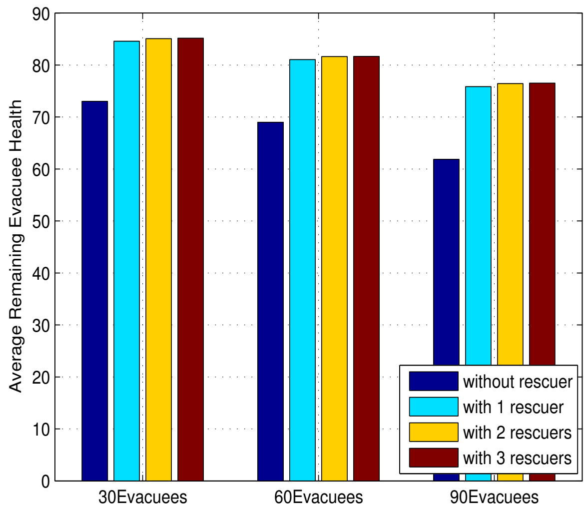


Figure 4.4: Average remaining evacuee health for each scenario.

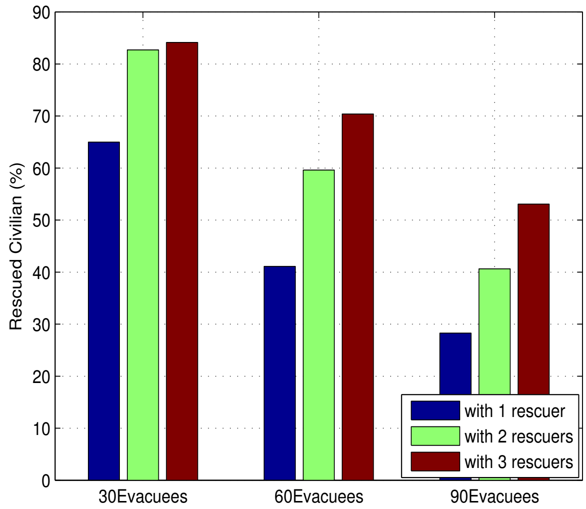


Figure 4.5: Percentage of evacuees who are saved by rescuers based on the total number of injured civilians for each scenario.

of the convergence of the curve is faster compared with the scenarios where introducing rescuers (see star points in Figure 4.2). This is due to the fact that it takes time to execute rescue procedure, thus, the evacuation procedure finishes slower in the situations with rescuers. Also, the evacuation time of the scenarios with 2 and 3 rescuers is shorter than the scenario with only 1 rescuer, which accords with our expectation that more rescuers can finish job quickly. Moreover, Figure 4.2 presents a higher percentage of civilians exiting the building with rescuers in the system compared with no rescuer in the system (also see Fig. 4.3), which confirms that the existence of the rescuers enhances the human outcome of emergency situation.

In Figure 4.3 and Figure 4.4, the overall results of the effectiveness of rescuers in the system are presented. In Figure 4.3, the bars show the average percentage of survivors without and with rescuers under different population density. As can be observed, in the scenarios where rescuers are used, the evacuation procedure is more effective with a higher percentage of civilians exiting the building. The simulations with lower density (30 evacuees) show that the system with two rescuers reaches a high performance that 97.59% civilians can exit the building with 95% confidence interval, however, adding one more rescuer (three rescuers in total) has no obvious effect on increasing the percentage of survivors since two rescuers are enough to saving injured civilians in the low density of population. As the occupancy level of the building increases (60 and 90 evacuees), the effectiveness of introducing more rescuers is more apparent (also see Fig. 4.5). The simulation results in terms of average remaining evacuee health are illustrated in Figure 4.4. Each civilian starts with an initial health of 100, which decreases based on the hazard intensity as the civilian is exposed to the hazard (i.e. fire and smokes) [8]. The rescuers carry trapped civilians towards the evacuation exits by protecting them from hazardous areas, achieving better results compared to the case where no rescuer is presented. Comparing with different number of rescuers under diverse population density, it can be seen that adding more rescuers cannot significantly improve the remaining health of evacuees. This is explained by the fact that when the fire starts all civilians navigated by the same strategy try to get out the building quickly, which cause the pedestrian congestion problem, while introducing more rescuers in the building will aggravate the congestion. Thus, both evacuees and rescuers need more time to move inside the building and the probability of exposing to the hazard increases. Figure 4.5 shows the percentage of rescued civilians based on the total number of injured civilians for three scenarios, with one, two and three rescuers separately, and the results are directly to describe the performance of introducing rescuers in the system. With low density of 10 civilians per floor, the system with 3 rescuers increases only 1.43% compared with the system with 2 rescuers. However, it is more obvious to see the effectiveness of adding 3 rescuers compared with 2 rescuers for higher population densities and the increased ratios is 10.78% and 12.43% for 20 and 30 civilians per floor. We can still verify the results obtained from Figure 4.3, that the improvement of importing rescuers is more obvious for high population density. However, comparing with three different scenarios, the percentage of rescued civilians decreases with the civilian density increasing since high population density leads to the congestion problem, causing more injured civilians. Therefore, an appropriate number of rescuers should be considered for different population densities, since an overestimate of rescuers may lead to resource wastage or more serious casualties due to the congestion problem.

4.2 Performances of RNN based algorithm in Rescue System

The effect of RNN based task assignment algorithm will be illustrated in this section, and the problem observed in the results of introducing rescuers in evacuation system will be discussed firstly. Section 4.1 has shown that the evacuation system with rescuers can significantly decrease the fatalities in emergency situations, however, there are three phenomenons can be noticed in the simulations and these phenomenons are obvious in high population density environments, (1) more than one injured civilians may be assigned to one rescuer, but the capacity of each rescuer is one so that there may be some victims who cannot being rescued due to the assignment problem; (2) more than one rescuers may be allocated to one injured civilian even though the first allocated rescuer is able to save the injured, and because of the limited number of rescuers some victims do not being assigned to any rescuer; (3) the rescuers may be injured during rescue and then cannot finish their tasks. The reasons for phenomenon (1) and (2) are that the allocations between rescuers and victims are treated as uncorrelated, which means that there is no relationship and effect between each assignments. In other word, the allocation of a rescuer to a victim does not consider the assignments have already done, thus, it may cause that the same rescuer is allocated to more than one victims and/or the same victim is assigned to more than one rescuers. In high population density environments, more evacuees are trapped and immobilitied at bottleneck points, such as stairs and evacuation exits, due to the pedestrian congestion problem, and then more people need to be assigned a rescuer simultaneously. Therefore, it is more possible that the rescuers receive more than one rescue messages from the sensor network and/or the same rescue message may being sent to more than one rescuers. In this case, the rescuers select their rescue target randomly, which leads to the problem (1) and (2). The result of (3) indicates that the rescuer may be not perfectly effective or suited to save a victim though this rescuer has been allocated. This is due to the fact that although the rescuers can reach at the emergency sites very quickly and they carry the protective devices, they may not know the inside situations in uncertain environments, for example, how the hazard intensity, and whether there is congestion, thus, it is possible that they expose to highly hazardous areas long time, since they toward the disaster areas and they may be encounter the congestion problem.

However, RNN based task assignment algorithm can alleviate these three problems by taking into account N fully connected neurons with a cost objective function, where $N = |R| \times |V|$, $|R|$ is the size of the set of rescuers and $|V|$ is the size of the set of victims. As mentioned in section 3.3.1, RNN based task assignment algorithm treats each rescuer-victim pair as a neuron, such as neuron i means that rescuer r is allocated to victim v, and the algorithm minimize the total execution 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, rescue failure probability of neuron i), and the penalty of not rescuing victim v. Therefore, with the cost objective function, a rescuer-victim pair will be selected if this allocation can reduce the total cost of rescue, for example, if the rescue failure probability of neuron i is high, this neuron may be not be selected since this allocation may be increase the total execution 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 allocation of distinct rescuers to the same victim and also avoid the same victim being assigned to distinct rescuers. Thus, the problem (1) and (2) can be avoided, to some degree. With RNN based task assignment algorithm, the emergency rescuers can take actions strategically in order to save more people out of danger with less response time.

The effectiveness of the proposed RNN based task assignment algorithm is evaluated in term 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 given sizes of the set of rescuers and the set of victims, and both the two metrics denote straightforwardly the efficiency of RNN based algorithm. The results of our experiments are obtained by comparing two different scenarios, without task assignment algorithm and with 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 hazard in the building. In simulations, rescuers will execute rescue task only once, since the rescuers are considered to be expendable and non-reusable that has been mentioned in Section 3.3.2. In algorithm 1, the parameters $K(v)$ for each victim in V are generated from the uniform distribution in the interval $[0, 50]$, and the cost of assignment $C(r)$ for each rescuer in R, is taken to be independent from its assigned victim and belongs in the interval $[0, 10]$. The rescue failure probabilities $q(r)$, is also taken to be independent from its assigned victim and are randomly generated in the interval $[0.05, 0.15]$. The excited probabilities of neuron (r, v) is initialized to zero for all neurons in the network. All RNN parameters are updated 20 times for each allocation to make convergence, and the neuron with the highest probability of being active will be selected. The results are summarized in Table 4.1 and Table 4.2.

Table 4.1: 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 4.2: Number of evacuees saved by rescuers

No. of rescuer $\lvert R\rvert$	No. of victims $\lvert V\rvert$	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

Table 4.1 shows that the rescue system with RNN based task assignment algorithm can increase the percentage of evacuees saved by rescuers based on the total number of injured civilians, and Figure 4.6 represents the data obtained from Table 4.1 using bar charts, which make the results visualized. In Figure 4.6, it is clear to see that the percentage of rescued civilians decreases with increasing the number of evacuees in the building, and this is due to the fact that it takes relative long time to evacuate in high population density emergencies due to the congestion problem, which then increase the time of exposure to the hazard and reduce the number of survivors. Without task assignment algorithm, the blue bars show that the rescue system with 7 rescuers decreases the percentage of rescued civilians compared with the system with 5 rescuers for each scenario (30 evacuees, 60 evacuees, and 90 evacuees). The reason is that when there are people who need help, the rescuers enter the building and toward hazardous areas simultaneously, even though they have known the right way to rescue and evacuate, and they could reach at the hazard very quickly, some of them may increase the congestion along with road and then the rescuers as well as the remaining evacuees may spend lots of time waiting or even be obstructed by congestion in road. As any spot may turn to dangerous at any time, longer waiting time means less chance to survive. Moreover, without task assignment algorithm, it is possible that the distinct rescuers are allocated to the same victim and/or the distinct victims are assigned to the same rescuer, so that, more rescuers may be not rescue more injured civilians. Therefore, if introducing more rescuers in the system where does not apply task assignment algorithm, it will not only aggravates the congestion problem, but also may be not increase the number of people rescued. In this case, the performance of the system with more rescuers may be worse.

However, with RNN based task assignment algorithm, the red bars show that the percentage of rescued civilian has a tendency to increase if introducing more rescuers in the system. Although the rescuers still increase the congestion in the building, RNN based algorithm makes the rescuers save victims effectively and quickly, which dominates the overall performance of the rescue system. Thus, more rescuers is able to save more people with RNN based task assignment algorithm, and then the percentage of rescued civilians increases. Comparing the blue bars with the red bars, we notice that the system

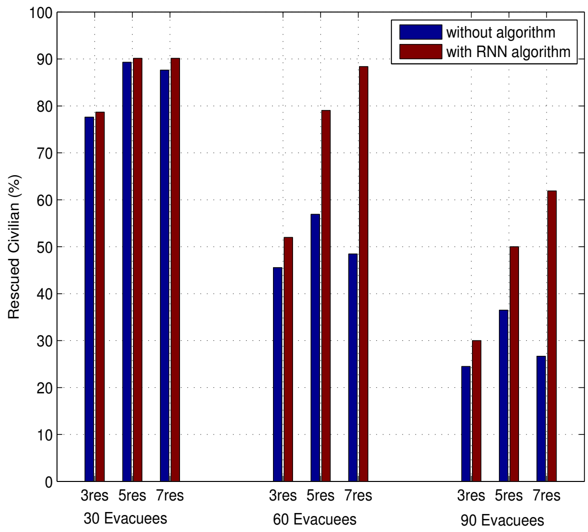


Figure 4.6: Percentage of evacuees saved by rescuers based on the total number of injured civilians.

with RNN based task assignment algorithm performs better, especially in high population density environments (60 evacuees and 90 evacuees).

In Table 4.2, given a set of victims, the increase of the number of rescuers cannot significantly improve the performance on the number of rescued evacuees without task assignment algorithm, and also more rescuers may be reduce the number of rescued evacuees, which confirms with the results obtained from Figure 4.6. However, the system with RNN based task assignment algorithm can provide a near-optimal performance, where

$$If \;\; |V| < overall\ maximum\ capacity\ of\ rescuers$$

$$number\ of\ rescued\ evacuees \overset{approx}{\longrightarrow} |V|$$

$$else$$

$$number\ of\ rescued\ evacuees \overset{approx}{\longrightarrow} overall\ maximum\ capacity\ of\ rescuers$$

$$end.$$

Here, the capacity of rescuers is one and the maximum capacity of rescuers is two, which have been defined in Section 3.1. Note that, if the injured civilians are far away from each other, the number of rescued civilians for each rescuer may be 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 along with toward exits and the capacity of the rescuer does not reach the maximum capacity, this victim will be carried by the rescuer. Therefore, if the locations of injured civilians are not close, it is impossible to select a road by the rescuer that will get through other injured locations.

Chapter 5

Conclusion and Future Work

5.1 Summary of Results

5.2 Summary of Problems

5.3 Open Issues and Future Work

Bibliography

[1] 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.

[2] ——, "Future research on cyber-physical emergency management systems," *Future Internet*, vol. 5, no. 3, pp. 336–354, 2013.

[3] 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.

[4] H. Bi, A. Desment, and E. Gelenbe, "Routing emergency evacuaees with cognitive packet networks," in *Proceedings 28th Annual International Symposium on Computer and Information Sciences ISCIS 2013*. Springer, 2013.

[5] E. Gelenbe, "Cognitive packet network," *U.S. Patent 6,804,201*, October 11 2004.

[6] E. Gelenbe, R. Lent, and A. Nunez, "Self-aware networks and qos," *Proceedings of the IEEE*, vol. 92, no. 9, pp. 1478–1489, 2004.

[7] E. Gelenbe, "Steps towards self-aware networks," *Communications of the ACM*, vol. 52, pp. 66–75, 2 2009.

[8] G. Gorbil and E. Gelenbe, "Opportunistic communications for emergency support systems," in *Int'l Conf. Ambient Systems, Networks and Technologies*, 2011, pp. 1–9.

[9] G. Gorbil, "Opportunistic communications for emergency support," *Thesis submitted for the Degree of Doctor of Philosophy of Imperial College London and the Diploma of Imperial College*, January 2013.

[10] E. Gelenbe and G. Loukas, "A self-aware approach to denial of service defence," *Computer Networks*, vol. 51, no. 5, pp. 1299–1314, 2007.

[11] S. Li, A. Zhan, X. Wu, and G. Chen, "Ern: Emergence rescue navigation with wireless sensor networks," *Parallel and Distributed Systems (ICPADS), 2009 15th International Conference on*, pp. 361–368, 2009.

[12] E. Gelenbe and S. Timotheou, "Random neural networks with synchronised interactions," *Neural Computation*, vol. 20, no. 9, pp. 2308–2324, 2008.

[13] 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*, October 2010.

[14] Q. Li, M. D. Rosa, and D. Rus, "Distributed algorithms for guiding navigation across a sensor network," in *ACM Int'l Conf. Mobile Computing and Networking*, 2003, pp. 313–325.

[15] 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, 2006.

[16] 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, 2006.

[17] M. Barnes, H. Leather, and D. K. Arvind, "Emergency evacuation using wireless sensor networks," in *IEEE Conf. Local Computer Networks*, 2007, pp. 851–857.

[18] 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.

[19] Y. Inoue, A. Sashima, T. Ikeda, and K. Kurumatani, "Indoor emergency evacuation service on autonomous navigation system using mobile phone," in *Int'l Symp. Universal Communication*, 2008, pp. 79–85.

[20] 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.

[21] 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, 2010.

[22] E. Gelenbe, E. Şeref, and Z. Xu, "Simulation with learning agents," *Proceedings of the IEEE*, vol. 89, no. 2, pp. 148–157, 2001.

[23] J. Smith, "State-dependent queueing models in emergency evacuation networks," *Transportation Research Part B: Methodology*, vol. 25, pp. 373–389, 1991.

[24] A. Desmet and E. Gelenbe, "Graph and analytical models for emergency evacuation," *Future Internet*, vol. 5, no. 1, pp. 46–55, 2013.

[25] U. Halici, "Reinforcement learning with internal expectation for the random neural network," *European Journal of Operational Research*, vol. 126, no. 2, pp. 288–307, 2000.

[26] E. Gelenbe, "Learning in the recurrent random neural network," *Neural Computation*, vol. 5, no. 1, pp. 154–164, 1993.

[27] 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.

[28] J. Reich and E. Sklar, "Robot-sensor networks for search and rescue," in *IEEE Int'l Workshop Safety, Security and Rescue Robotics*, 2006.

[29] 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.

[30] “The foundation for intelligent physical agents (fipa),” http://www.fipa.org.

[31] D. W. Pentico, “Assignment problems: A golden anniversary survey,” *European Journal of Operational Research*, vol. 176, pp. 774–793, Jan. 2007.

[32] S. H. Bokhari, *Assignment Problems in Parallel and Distributed Computing.* Kluwer Academic, 1987.

[33] E. Gelenbe, “G-networks with triggered customer movement,” *Applied Probability*, vol. 30(3), pp. 742–748, Sep. 1993.

[34] 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.

Appendix A

Appendix