An improved pathfinding under multiple exits with SOA in a double-layered MANET

Jae-Suk Lee and Sung-Bong Yang

1Mobile Communication Lab, LG Electronics, Seoul, 129-24 Korea E-mail: wizsuk@gmail.com
2Department of Computer Science, Yonsei University, Seoul, 120-749 Korea E-mail: yang@cs.yonsei.ac.kr

Most of controlling disaster managements focus on how to rescue people from disaster areas. In this paper, we propose an emergency escape system that helps people escape by themselves effectively from a disaster area in which each person owns a mobile device. We define a scenario for the proposed system in which there are five system states; they are Idle, Discovery, Subscription/Publication, Execution, and Communication states. The system uses several technologies such as service-oriented architecture, GPS and Wi-Fi positioning systems, and some sensors. The proposed system utilizes users’ information such as weighted distances, weighted directions and revisit counts. The system is implemented on a double-layered MANET to reduce traffic and to collaborate with each other. We have constructed a simulator to test the proposed system. The simulation results show that the proposed system outperforms other approaches in terms of the distance and the time proving that the proposed system could be an effective life-saving rescue system.

Keywords: Web-service; Rescue system; Disaster management; MANET; Context-aware service; Path finding; Localization

1. INTRODUCTION

Controlling disaster management is a very serious matter since peoples’ lives are placed in danger. Recently various efforts have been made for developing disaster management systems [1]. Typical systems require a person within a disaster area to take a video shot of the current situation using a mobile phone and then to send the video clips using a video transfer system. The carrier then provides this video clip to other subscribers to inform and avoid the emergency situation. The aforementioned systems may not be effective when people need to find the exits within a disaster area, since the systems generally focus on informing the emergency situation to the people and then on the rescue and recovery from outside. Researchers have also studied methods for rescuing people using robots and for disaster recovery as management activities [2, 3].

In this paper we propose an effective escape system that allows people to rescue cooperatively using their mobile devices. The proposed system helps people find the exits using the information such as weighted distances, weighted directions and revisit counts. The system is implemented with the help of a few technologies such as service-oriented architecture to inform about the emergency and exchange information, GPS and Wi-Fi positioning systems (WPS) to locate mobile devices, and a mobile ad-hoc network to exchange the information among mobile devices to configure the states appropriately. We also propose a double-layered mobile ad-hoc network (MANET) to reduce the traffic and to let people communicate each other collaboratively. While searching for the exits, nodes (peoples) gather the information and use sensors like accelerometers and gyroscopes to know the directions to move. The rest of this paper is organized as follows. Some related works on the technologies used are provided in Section 2. The proposed system is described in Section 3. The experimental environment and results are presented in Section 4. Finally, we conclude the paper in Section 5 with the future work.
2. PREVIOUS WORKS

Many researchers and governments worldwide have made serious efforts on emergency managements. In this section, we discuss several methods related to control disaster and technologies to be considered in this paper.

2.1 The Method using Multiple Robots System in Disaster

Researchers have studied methods for rescuing people using robots and for disaster recovery as management activities [2]. The disaster area is generally dark and is filled with toxic smoke. The network infrastructure broke down. Therefore, it is extremely difficult for the rescue workers to find the people in the disaster areas. To overcome this, researchers have studied the methods exploiting robots. As shown in Figure 1, the tiny robots are deployed in a disaster area and are organized into triangular cells in a MANET. They may act as routers or nodes. If there are obstacles or blocks, the tiny robots are supposed to detect them. When the rescue workers enter the disaster area, they help let them know where obstacles and blocks are. Hence the rescue workers can locate more easily the people.

2.2 WORKPAD & MOBIDIS

When a disaster occurs, many rescue workers from the different organizations have difficulties in working cooperatively since they may have different systems of approaches with heterogeneous devices. WORKPAD is a European research project to aim at building and developing an innovative software infrastructure for supporting collaborative work of human operators in emergency/disaster scenarios [3]. The objective of the WORKPAD project is to investigate how to create communities and how to integrate them, and how to enable mobile teams to exploit such a back-end through the interplay of MANET technologies and workflow management. MOBIDIS is a middleware supporting workflows of a front-end equipped with mobile devices. In WORKPAD and MOBIDIS as shown in Figure 2, the working groups are divided into a back-end and a front-end. A back-end is the group connected a web-server and so on. A front-end is the group consisting of rescue workers. Each front-end organizes the leader and its members. Some members take pictures of critical spots in the disaster area. Others are gathering the information of the disaster area such as the building status and exits status. The leader is reported by its members. All communications among them are established by MANET because the infrastructure is assumed to be broken down. If some groups are from the same organization, they might have the common devices. But in the disaster area, the different organizations usually take part in rescuing. Therefore, their devices must be heterogeneous, and they may hardly communicate each other. The leader of each group is gathering the information from its members and is reporting to the back-end.

2.3 Transferring Video and Call Function to inform Emergency

In the event of fire or other disasters, people normally make emergency phone calls, waiting for the rescue or trying to find a way to escape from the dangerous area. If these people are not familiar with the disaster area, finding the way out becomes extremely difficult for them. Researchers and governments proposed several systems to control the emergency situation effectively [45]. Some systems require a person within the disaster area to video and record the situation using a mobile phone and then to send the video clips using a video transfer system.

The carrier then provides this video to its other subscribers to inform and avoid the emergency situation. Some other systems utilize the emergency function of a mobile phone. In this system, a phone number is registered and notified in the event of a disaster situation. An emergency call is made or the SOS function of the mobile phone is used.

The aforementioned systems are not effective when people need to find an exit from a disaster area. That is, the previous systems are focused on informing the emergency situation or rescue and recovery. Although a rescue team can immediately arrive to the disaster area after the information has been received, an active way to search for an exit utilizing mobile devices would prove to be more effective. People almost always carry their mobile devices with them, thus the proposed system utilizes mobile devices to actively search for an exit during disaster situation. We have considered several technologies for the implementation of this system.

3. RELATED WORKS

3.1 SOA and Context-Aware Service

Service Oriented Architecture(SOA) is designed to use services through networks without concerning about differences in interfaces and architecture. Web service is one of the methods involving services and uses XML-based messages. The data such as maps can be exchanged among heterogeneous devices. Figure 3 shows the basic Web service architecture, in which a service requester and a service provider interact based on the service description information published by the provider and discovered by the requester through some form of discovery agency [6].

XML messages compliant with the simple object access protocol(SOAP) specification are exchanged between the requester and provider; SOAP is a lightweight protocol intended for exchanging structured information between peers in a decentralized, distributed environment and provides the definition of the XML-based information. The provider publishes a web service description language(WSDL) file that contains a description of the message and endpoint information to allow the requester to generate the SOAP message and to send it to the correct destination; WSDL is a XML-based language for describing Web Services such as how to access them and where they are located. But the architecture in Figure 3 is not applicable to MANETS.

If service provider $S_1$ is accessible to service requester $R_1$ and discovery agency $D_1$ is not as in Figure 4(a), $R_1$ receive the service from $S_1$ but cannot discover the service list published by
Observe that a similar problem occurs when $R_2$ cannot access $S_2$ as in Figure 4(b).

Hence, we allow peers to play multiple roles of both a service provider and a discovery agency to overcome such a problem as in Figure 5. In the peer scenario, peers having web service instances such as service providers, service requesters, and discovery agencies collaborate with each other. Such architecture was provided in [7] and can be used to exchange information among peers.

3.2 Context-Aware Service

To inform the people in an isolated area during an emergency situation and to let their mobile phones guide them toward a closest exit, we considered a context-aware service as a web service; a context-aware service is to offer users a service relevant to their current locations. To provide a context-aware service, the devices should discover the service after connecting to the server. After receiving the service list offered by the server, a device may subscribe and select a service. The server notifies all subscribers when publishers post events such as detecting the fire and toxic smokes [8]. WS-notification is a family of related specifications that define a standard Web service approach to notification using a topic-based publish/subscribe pattern [9].

Figure 6 describes the operation among the publisher, subscriber, and broker based on WS-notification. A subscriber requests the list from the broker which is the web server. After getting the topic list, it subscribes to some topics which will be served. The publisher notifies the broker of the events. The broker informs the subscriber about these events. This technology can be used to inform the people about the emergency when a disaster is detected in our system.
3.3 Localization

Navigation can be either behavior-based or map-based. In a behavior-based navigation, moves are based on events, while moves are based on a map in a map-based navigation [10]. Localization is a method to locate people on a given map. People within a disaster area need to know their current locations to search the closest exits. A positioning system likes GPS or WPS can be utilized for the current locations. WPS uses the access points(APs) for indoor or building areas. The Skyhook wireless company provides WPS and developed an iPhone application [11]. After the current position of a mobile phone is known, the mobile phone performs the localization; that is, it builds a map and then identifies its position on the map.

We need to use some sensor technology to determine the direction and to locate dangerous areas such as fire and the presence of toxic smokes. Since it is not surely dependable nowadays, we can use geomagnetic, accelerometer and gyroscope sensors to know proper directions. Most smart phones such as iPhone have equipped with them.

3.4 MANET

Over the last decade, MANETs have been a challenging topic in wireless researches [12]. In this paper, we use a MANET to exchange information in the absence of a communication infrastructure. In a disaster area, an infrastructure is assumed to be
non-operational; fires burn the servers and the base stations are broken down. In such a situation, a MANET is a rapid establishment of communication infrastructure wherein mobile devices act as nodes and/or routers. A node can communicate with other nodes through the nodes acting as routers.

Figure 7 shows a double-layered peer-to-peer (P2P) system called the MIS (Maximal Independent Set) P2P system [12]. This system successfully reduces the network traffic over the ORION (Optimized Routing Independent Overlay) system. Note that ORION uses flooding as its communications; a node sends messages to all its neighboring nodes and such broadcasting cause enormous network traffic. Nodes in MIS send messages to only special nodes to reduce the network traffic. Special nodes are called super peers. A super peer manages some neighboring nodes, called sub-peers, and has all information about its sub-peers. In Figure 7, a random number is shown next to each node. A super peer is selected if it has a larger number than its neighboring nodes.

3.5 Pathfinding

Pathfinding describes the way to find a good path from the starting position to the destination. If there is a node on the map, it finds a path to the destination avoiding obstacles with minimizing costs such as time, distance and so on. Many researchers have studied algorithms to find a way to an exit using such as DFS, BFS, and A* search.

An example of a path finding algorithm is the Dijkstra’s shortest path algorithm which begins with a start position and puts adjacent positions to the ‘open list’. The position in open list with the shortest distance from the start position is examined. The position is marked in the ‘closed list’. This process is repeated until adding the destination to the closed list; the shortest path has been found if the destination in is the open list, otherwise the destination is not reachable.

A* search is a variant of the Dijkstra’s algorithm. It adds heuristic which is an estimated distance from the start position.
to the destination. A node moves along with $F$, where $F$ is the sum of $G$ and $H$; $G$ is the movement cost to move from the start position to the destination and $H$ is the estimated movement cost to move. If $H$ is 0, $A^*$ search is equivalent to the Dijkstra’s algorithm. Adjacent positions are added to the open list and a position with the lowest $F$ value is marked in the closed list. The process runs in the same manner as in the Dijkstra’s algorithm.

When a single node finds a path on the map and there is one destination on the map, $A^*$ search is a preferable search method. But when multiple nodes move at the same time and there are multiple destinations on a map, there are some problems [13]. We need not to search the entire area but to find the goal—an exit—from the current position. Normally, there are multiple exits in the area. Therefore, the proposed system should find an escapable exit. Note that searching is done during a disastrous situation and is different from the case when robots scan the entire area and then find the destination since some exits could already be collapsed. Thus, in this paper we propose an effective way to find escapable exits.

4. THE PROPOSED SYSTEM

We assume that the infrastructure in a disaster area had been broken down and only a web server and several APs are working. People within the disaster area have mobile phones with the emergency modules to operate the emergency tasks such as subscribing to the web services, calculating a way to a closest exit, and communicating with other nodes in a MANET.

With these assumptions, we propose an effective escape system to find the way to an exit. The proposed system offers the people isolated in a disaster area to escape independently before an emergency rescue team arrives. It is also possible to provide an emergency team with the information on the status of the disaster area. Since a disaster area is generally dark and people are often not familiar with the area, they tend to spend more time in finding a way out. Thus, this proposed system is an effective way to provide a solution for such a case.

The proposed system consists of five states; the idle state, discovery state, subscription/publication state, execution state and the communication state. In the discovery state, the services provided by a server are detected. In the subscription/publication state, the user is offered of the service list and allows the publishers to post the events such as a sensor detecting fire, smoke, among others. The server informs the subscriber about the emergency. In the execution state, the mobile phone notified of the events executes the emergency module to find a path and communicates to other subscribers. If other subscribers are within the communication range in MANET, they exchange proper information in the communication state.

4.1 A Scenario

We now define the architecture of a mobile phone and scenario. As shown in Figure 8 Emergency module (1) is an application running on a mobile phone which offers visual guidance to the user to find the exits. Mobile web server(MWS) (2) is a module in charge of acting as a server and offering a service list to its clients using SOAP messages (3). In Figure 8 (4), base-bands and sensors are to provide communication and direction, respectively.

In the proposed system there are five states as shown in Figure 9.

4.1.1 The Idle State

In the idle state people use mobile phones for their own usual purposes—they are not in emergency. They can surf the internet using browsers, make phone calls, and so on.
4.1.2 The Discovery State

Before a disaster occurs, the discovery state is entered when a mobile phone user moves into a place where a special service is offered. People’s mobile phones dynamically discover the services from the web server. Mobile phones act as subscribers, the web server acts as a broker, and the mobile phones and sensors such as fire detectors can be publishers.

Note that the terminologies like subscriber, publisher, and broker are taken from the WS-notification specification in [9]. The mobile phones receive the map of the disaster area including the information such as the walls and the exits. We can determine the map before a disaster takes place in a building, since all buildings have their own blueprints. If there are any other events, the publisher notifies the subscribers of these events.

4.1.3 The Subscription / Publication State

The sensors acting as publishers inform the server about the emergency status during fire or a building collapse. In other methods, if a person is out of server’s communication range, a person presses a designated emergency button of the mobile phone or selects the emergency module (Figure 8(1)). The mobile phone acting as the publisher reports the events to the server. If even server broke down, mobile phone acts as a server through MWS when another one gets in communication range.

4.1.4 The Execution State

The server notifies the subscribers—the mobile phones—about the emergency events. The transmitted events invoke the emergency modules of the mobile phones for escape or rescue. The mobile phones have already received the map on the disaster area. But we cannot determine the dangerous areas until disasters break out. An exit is searched while avoiding dangerous areas. For searching closest exits, we divide the map into grids based on the communication scope of mobile phones. A person with a mobile phone moves from a grid cell to another with visual guidance on it. A person holding the mobile phone is called a node. Thus, we model the real world as the nodes on the grids.

After that, we need to localize each node; that is, we find the current position of each node on the map. WPS is used to get the current position in an indoor environment. We need to calculate a best possible way to the closest exits. While searching for the exits, nodes gather the information, use the sensors such as accelerometer and gyroscope to know the directions to move (Figure 8(4)) and save the information to their memory during the execution state.

4.1.5 The Communication State

If a node is within the communication range of other nodes, they can exchange the information about the location of the dangerous areas and collapsed exits. If an exit is not found, the search process and information exchange are repeated.

The process of the communication follows in detail between mobile phones A and B in Figure 9

• A wants to communicate with B
• A invokes B’s mobile web server.
• B offers a service list to A through WSDL

The service lists can be one of the lists: IsnodeType, synchronizationInfo, IsSuperPeer relay (sends requested node’s message to the other super peer) etc

• A selects one of the services and sends the message to B through SOAP.
• B processes A’s request and responses the result back to A.
4.2 An Effective Search Algorithm

We now focus on the execution state in the proposed scenario for describing an effective search algorithm. Dividing the map into grids simplifies handling and calculating the distances of nodes. A grid cell may be marked by a block or an exit.

When a node tries to move to the closest exit, we measure the distance between the node’s current location and the exit; we call the distance mileage. Observe that the shorter the mileage is, the better the system performance we get. In order to evaluate the system performance, we let a node have the $x$- and $y$-coordinates and an angle for its direction; the direction is one of the eight directions defined in Figure 10.

A node is assumed to move from the center of a grid cell to that of the next until an exit is reached. Hence we measure the Euclidean distance $d$ between the two centers for the distance between two adjacent cells. In Figure 10, the initial coordinate of the node is $P_0(x_0, y_0, \theta)$. Then we can get the coordinate of each of eight adjacent cells. If the node moves from $P_0$ to $P_1$ or $P_2$, its distance is $d = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}$ or $(x_2 - x_0)$, respectively. The node’s movement will be accumulated to its mileage as the total distance it moved. It can be seen that a higher mileage of a node induces that it takes longer time in searching an exit.

We now propose a search algorithm for the proposed system. We call the proposed system the weighted direction system (WDS). We also define other systems for comparison; these systems are the random direction system (Rand), the clockwise direction system (CL), the counter-clockwise system (CC), and the user-defined system (UD).

4.2.1 Weighted Direction System

In WDS, we divide the map into the four quadrants with respect to the node’s current position as the origin. Then a value called weight is calculated with the equation below to decide which direction it should move.

$$weight = distToExit + dirToExit + revisitCount,$$

where $distToExit$ is the hop count from the current cell—in which the node is currently located—to the closest exit, $dirToExit$ is the direction value with respect to the closest exit, and $revisitCount$ of a cell represents the number of times that the cell visited. Note that $revisitCount$ of each cell is 0 initially and is increased by one for each revisit to the cell.

We describe each term using an example in Figure 11. In Figure 11 (a) a map received from the server is given. The map in (b) shows after a disaster occurs; the blocks and dangerous cells are added to the map. The numbers in the cells of the maps in (c) and (d) are $distToExit$ with respect to the two exits, respectively.

In determining $dirToExit$ for the node with respect to the exit in the top-right corner, we assign 1 to $dirToExit$ if the node tries to move to the exit, assign 4 if it should move to the opposite direction of the exit, and assign 2 or 3 if it moves to either left or right direction with respect to the exit. Hence, it can be viewed that as if we divide the map into four quadrants based on the node as the origin as shown in Figure 12. Therefore, $dirToExit$ to the first quadrant (1Q) is 1 since the exit is in 1Q, $dirToExit$ to 3Q is 4, $dirToExit$ to 2Q and 4Q are 2 and 3, respectively. Note that if the node moves to another cell, it decides $dirToExit$’s value every time after checking $distToExit$’s value of the exits from the current cell.

We now explain how the search is performed in WDS with the example in Figure 12. We assume that $revisitCount$ of each cell is zero before the search and the search in WDS prefers a diagonal direction to an orthogonal one when both cells have the same weights.

First, as shown in Figure 12(a), the node selects 1Q in which the top-right exit is located and $dirToExits$ of 1Q, 2Q, 3Q and 4Q are 1, 2, 4 and 3, respectively. It tries to move to the cell in
NE since the weight of the cell is $1 + 1 + 0 = 2$ and then to the cell (its weight is also 2) in E, but both are ‘dangerous’ cells. The node now can move to N.

As in Figure 12(b), the node selects 4Q in which the closest exit is located from the view point of the current cell (in case when an exit lies in between 1Q and 2Q, we regard it as if it belongs to 1Q, and when it lies between 2Q and 3Q it belongs to 2Q, and so on.) and dirToExit of 1Q, 2Q, 3Q and 4Q with respect to the current exit are 2, 4, 3 and 1, respectively. It tries to move to the cell (its weight is $1 + 1 + 0 = 2$) in SE and then to the cell (its weight is also 2) in E, but both are ‘dangerous’ cells. The node can move to the cell (its weight is $2 + 1 + 1 = 4$) in S, but the cell has been visited.

Hence the node selects 1Q next as in Figure 12(c). It tries to move to NE but there is no cell to move and then tries E, but it is a dangerous cell. It tries to move to N, but there is no cell to move. Now the node searches 3Q next as in Figure 12(d). It tries to move to the cell in SW, but it is a dangerous cell. It moves to the cell (its weight is $3 + 3 + 0 = 6$) in W, although the cell in S has the same weight; the cell in S was visited before.

Note that a node computes distToExit with respect to the exits from a newly moved cell before checking dirToExit. Hence it calculates distToExit with respect to both exits and finds out Exit 1 is the closest one as in Figure 12(e). The node now searches 3Q and dirToExit of 1Q, 2Q, 3Q and 4Q are 4, 3, 1 and 2,
respectively. It moves to SW and finally the node finds Exit 1 as in Figure 12(f).

4.2.2 Other Systems

RD decides the direction to move randomly. A node moves to a cell in one of the eight directions equally likely. This system is to simulate people wandering flustered. When a disaster happens, most people are so frustrated that they tend to go any direction to find the exits. CL determines the direction to move clockwise. A node gets start moving a specific direction; in our case, start direction is E. If there are dangerous and block grid at the direction to move, a node changes the direction clockwise. CC is \{E, SE, S, SW, W, NW, N\}. CC decides the direction to move counterclockwise. This system is the same as CL except changing the direction counter-clockwise. UD is \{E, NE, N, NW, W, SW, S, SE\}. UD decides the directions based on a predetermined order. For the experiments, we decide the direction as \{W, N, S, E, NW, SW, NE, SE\}. That’s because UD firstly searches the exit in the four cardinal directions.

In each system, a node also moves to an adjacent cell based on revisitCount to avoid a return visit.

4.3 Effective Collaboration using Double-Layered P2P in MANETs

We now depict how to organize a MANET in a disaster area. We assume that nodes may have different movements for searching exits based on CC, CL, UD, RD and WDS and the information is gathered by some sensors. The nodes within their communication range can share information on the current situation in the disaster area.

Figure 13 shows an example that other nodes may receive valuable yet critical information from other node who had experienced failure in searching for some exit(s) so that they do not have to go through the failure. In Figure 13(a), node \(N_1\) tried and found that the exit in 4Q is of no use. While returning its start position, node \(N_2\) meets \(N_1\); \(N_1\) sends the information regarding the ‘troubled’ exit using MANET. Hence, \(N_2\) goes to the east, reducing the mileage and the time to find other exit.

We propose the network models to compare performance in terms of time, mileage and network traffic. The network traffic means how much the information such as dangerous, block and so on are exchanged with each other when nodes are within transmission range. It is important to consider network traffic in MANET because of low power and bandwidth of mobile phone.

Figure 14 shows how each node exchanges information. As shown in Figure 14(a) nodes are connected in an unstructured manner so that the traffic on the network can be quite heavy due to flooding mechanism for message broadcasting. We call this model the Everybody-Meet model(EM).

On the other hand, as shown in Figure 14(b), the nodes are classified into super peers and sub-peers: super peer manages zero or more sub-peers. Each super peer has the appropriate information of its sub peers. When a node needs to exchange information, the request is sent to its super peer. We assume that super peers are the people who know the area well like the employees who work there; they know where the exits are so that they move with WDS. If node A moves into the communication range of super peer B, A notifies its presence to B and becomes a sub-peer as in Figure 14(b). D can send information B via C, while B and F can communicate each other through both C and E. We call this model the Employee-Customer model(EC).

5. ASSESSMENTS

In this section we evaluate the proposed system WDS and other systems in terms of mileage and time. For the MANET models, we use EM and EC as well as a model called No Policy(NP). In NP model, no node helps each other; each node tries to survive alone. We have built a simulator to perform the experiments on various maps. As shown in Figure 15, we can also set up each system to find an exit for a given node and can identify the direction of each node on the window of a mobile phone as the guidance using an arrow along with other information.

The simulator divides the disaster area into grids based on the transmission range. A grid cell has a color to reflect the status of the cell; if it is red, the cell is a dangerous area, blue for an exit, and black for wall. A node can move to an adjacent “white” cell or to an exit. Note that a user may come back to its initial position as well. Eight positions including the current position can be sent to other nodes. Nodes start to move from their own initial positions according to policies of the systems mentioned in the paper. The simulator also provides three network models so that each node behaves based on a different system to organize a MANET with one of the three network models.

The experiments were performed with 20 different maps for each system. The experimental parameters are given in Table 1.

We assume that the area is on a floor in a building, there are 20 nodes with 80m communication range, and there are up to four exits. First, each system divides the area into grids with respect to the communication range; that is, a cell is a square of 80mx80m.

We tested each system to evaluate the average of mileage and time. The sequences of directions for CL, CC, and UD are \{E, SE, S, SW, W, NW, N\}, \{E, NE, N, NW, W, SW, S, SE\}, and \{W, N, S, E, NW, SW, NE, SE\}, respectively. In the sequence of UD, we first try to search directions among N, E, S, and W and then go over a random order of the diagonal directions. The performances of the systems are compared in Figure 16.

Figure 17 shows various maps to perform the experiments.

There were wide deviations in the performances from map to map on CC, CL and UD, since these three systems heavily depended on the locations of exits and the distributions of blocks and dangerous cells in the area; Figure 18 supports such results. On the other hand, Rand and WDS were shown ‘stable’ results.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of maps</td>
<td>20</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>20</td>
</tr>
<tr>
<td>Communication range</td>
<td>80m</td>
</tr>
<tr>
<td>Area</td>
<td>2,400 x 2,240m²</td>
</tr>
<tr>
<td>Number of exits</td>
<td>4</td>
</tr>
</tbody>
</table>
for all the maps. The average mileage and time of WDS were improved 1,690% and 2,415% over UD that showed better performance than the rest. All systems except WDS searched the entire area with only directions and avoiding revisiting grid cells. On the other hand, WDS outperformed other systems because of its divide and conquer search method in which the whole area is

Figure 12 An example of a node's movement.

Figure 13 Example of exchanging information using MANET.
divided into smaller ones along with distances and directions every time a node moves into a cell. Note that WDS also considers the revisit count.

The reason why other systems excepting WDS have different results is that they heavily depend on directions and locations of blocks, exits and dangerous grids. Figure 19 verifies these results; the black dots are nodes, the cross marks (visited cells) were propagated toward the arrows, respectively. Nodes were propagated from 3Q and 4Q in Clockwise, 2Q and 3Q in Counter-Clockwise, 2Q and 3Q in User-Defined, respectively. On the other hand, random direction system and WDS were shown similar results on different maps, respectively. Note that since a node in WDS moves to the nearest exit with respect to its current location no distribution chart is given.
Figure 20 supports such results. We experimented maps which quadrant exit is; they are located on four quadrants of the map evenly: exit-1 is in the 1Q, exit-2 is 2Q, exit-3 is 3Q and exit-4 is 4Q. We experimented in the order named adding exits. As shown in Figure 19, the better performance for clockwise, counterclockwise and user-defined was represented in 3Q and 4Q, 2Q and 3Q, and 2Q and 3Q respectively. These results exactly corresponded to the distribution chart of Figure 18. In case of random direction system, even if exit was in which quadrant on the map, there was a narrow variation. WDS was also shown as a narrow variation. Moreover, it was the best performance. It shows us outperformance regardless of locations of the exits and another factors.

We also tested each MANET model to evaluate the performance in terms of mileage and time as well as the network traffic, since in the mobile ad-hoc environment network traffic is so important. We measured the number of maps exchanged among all the nodes during the simulation as the network traffic amount. For all MANET models, first we randomly assigned a different system to each node so that the expected number of nodes with each of the systems—CC, CL, Rand, UD, and WDS—is the same. We also controlled the minimum number of nodes with a system.
so that there are at least three nodes with the same system for balancing the environment.

Each node assigned to WDS should move according to Rand until the node meets other nodes, since the nodes moving with WDS are much faster than the nodes with other systems as shown in the experiments. Such actions can be viewed as if the employees look for the customers when a disaster takes place. Figures 21, 22, and 23 show the expected mileage, time, and network traffic for three MANET models.

The average mileage and time of EC were improved 16.0% and 16.2% over EM, respectively. The average traffic of EC was improved 494.1% over EM. The results of NP were the worst since nodes didn’t share the information achieved from their wanderings, while the results of EM showed better performance than NP in terms of mileage and time since nodes share their experiences with others to avoid the cells that no one should visit. But because such information exchanges cause EM to get the highest network traffic. EC showed the best performance...
overall, since each node with WDS played a super peer role to lead other nodes toward the exits. Regarding network traffic, EC was better than EM since information exchanges were processed mostly among super peers.

6. CONCLUSION

This paper has examined three aspects of disaster management: a definition of a scenario in disaster, an effective system to find exit and an effective network model to collaborate. We have defined a scenario to control disaster, provided systems including search algorithms, presented network models and showed the experimental results.

6.1 Summaries of the Solutions to the Research Issues

In this paper, we defined a scenario to escape from a disaster area for the proposed system in which there are several states: Idle, Discovery, Subscription/Publication, Execution and Communication states. In the proposed system, the direction and distance from the closest exit and the revisit information to a cell are used to help people with a visual guidance toward the closest exits effectively. We also defined the EC MANET model to reduce traffic, time and mileage for simulating more realistic environments. The experimental results showed that the proposed system outperformed other systems significantly and the proposed MANET model is more effective in terms of traffic, time and mileage. While most of disaster management systems
6.2 Future Work

We will implement our system to actual targets such as android mobile phones or iPhones. Although some sensors such as detecting fire and toxic smoke is not implemented yet on the devices, we expect that such sensors are implemented in the near future and therefore we can also use other functions like pressing or touching the screen.

We plan to study further on MANET protocols that can be applied to the proposed system and perform the simulation on the Network Simulator NS3 for future work.

Most of emergency escape systems consider buildings and subways stations which have many stories. Our proposed system can be extended to such structures with multiple floors. After receiving the blueprints of the disaster area with multiple floors,
Figure 23 Average traffic to find exit on 20 maps.

Acknowledgment

This work has been supported by the Korea Science and Engineering Foundation (KOSEF) for the research (2010-0015846).

REFERENCES


