Dispatching and Routing of Emergency Vehicles in Disaster Mitigation using Data Fusion ⋆

Arun Jotshi [a] , Qiang Gong [b] and Rajan Batta [a,*]

[a] *University at Buffalo, Department of Industrial and Systems Engineering, and Center for Multisource Information Fusion, 420 Bell Hall, Buffalo, NY 14260, U.S.A.*

[b] *Corporate Research and Development, United Airlines, Elk Grove Village, IL 60007, U.S.A.*

Abstract

The aim is to develop a robust methodology for dispatching and routing of emergency vehicles (EVs) in a post-disaster environment with the support of data fusion for decision-making. In this work we consider an earthquake scenario with a large number of casualties needing medical attention. With the influx of information (about the casualties, road and traffic conditions, etc.) data is fused to give estimates about the entities under consideration. We use this information to dispatch and route ambulances to casualty pick-up locations and then delivery to appropriate hospitals. The key factors that affect the dispatching of ambulances to a patient location include− patient priority, cluster information, and distance. Similarly, those affecting the dispatching of ambulances from a patient location to a hospital include− waiting time estimates at hospital emergency rooms, hospital capacities, and distance. Routes need to be generated for these ambulances taking into account the large scale of the real-world road network in addition to road damages, congestion and other such factors effecting routing. We present a dispatching and routing simulation model and a case study to show the performance of the developed methods.

Key words: Emergency Response Services, Data Fusion, Dispatching, Routing, Simulation, Large Scale Networks.

1 Introduction

In case of an emergency situation, data consisting of reports about casualties, structural damage, road conditions and traffic situation etc., begins to flow from the original site to various Emergency Response Centers (*ERCs*). The data flow becomes much more complex

⋆ This research is supported by the Air Force Office of Scientific Research (AFOSR), Grant number F49620-01-1-0371.

* Corresponding Author
Email address: `batta@eng.buffalo.edu` (Rajan Batta).
URL: `www.acsu.buffalo.edu/ ˜ batta` (Rajan Batta).

in case of a large scale disaster, given the amount of information (false or accurate) that is coming in. The science of efficiently organizing and interpreting such massive amounts of data is called *data fusion*. It appeared in the scientific literature in the late $1960s$ and it has been implemented in multiple disciplines since the $1970s$. A typical situation is one in which information flowing from multiple sources has a highly variable character. It is necessary to align the incoming data and assess the situation accurately and quickly so that proper action can be taken in a timely fashion.

We consider an earthquake as our dynamic post-disaster environment in which we have, in addition to the first impact, aftershocks that cause additional effects. In the immediate aftermath of an earthquake, ERCs might have to deal with collapsed buildings, fires, and hazardous material spills. Management of emergency service resources in such an environment requires efficient dispatch and routing strategies that provide rapid response for casualty pick up and delivery. The goal is to service the maximum number of the highest priority casualties with minimum service times. The information of casualties and the status of the roads is reported by sensors. Sensors consist of satellites, sensor systems embedded in the infrastructure (e.g. surveillance cameras), police, civilians and reports (phone calls) received at the ERCs. Information begins to flow from multiple sources to the fusion center. The variability of the sources leads to imprecision of the raw data which makes it difficult to determine the precise situation. After original data has been collected, the Data Fusion Center (DFC) fuses this raw data and reports the fused data estimates to the Dispatcher-Router (DR) center.

Information on each patient is composed of his/her location and injury class. We classify casualties into four priority classes: type 1 (mildly injured), type 2 (moderately injured), type 3 (severely injured), and type 4 (mortally injured). These casualties are distributed randomly in space and if a certain number of casualties are concentrated in a particular region, they form clusters. Roughly speaking, a cluster can be thought of as a small geographic region where a certain number of casualties are aggregated together. Information on each link (road segment) constitutes the level of damage and the congestion on the link during a particular interval of time. Each level of damage associated with a link has a probability associated with it, which is the probability of this information being accurate. In case of a major disaster the discovery rate of casualties is very high. A queue is created in which the casualties wait for service by an Emergency Vehicle(EV). Hence the need for an efficient dispatching and routing strategy.

The structure of the paper is as follows. In section 2 we define the basic infrastructure of the earthquake simulation model and the role of the DR federate in the simulation model. Section 3 is dedicated to literature review about data fusion and the field of emergency response (ER) services in general. Section 4 explains the elements of data fusion that we utilize. Section 5 explains the dispatching methodology developed to dispatch EVs for casualty pickup and delivery. Section 6 explains the routing methodologies developed taking into consideration the real time road data about damage and congestion. We present a case study in section 7 along with the results obtained by using the developed methods. Conclusions and future work are presented in section 8.

2 Simulation model

CMIF is the home of a multi-year program to address Information Fusion in support of crisis-center decision-makers dealing with post-event situations for both earthquakes (a natural disaster) and chemical attacks (a man-made disaster). The thrust of this simulation at CMIF is to provide realistic data that simulates the chaotic flow of both information and misinformation in the immediate aftermath of a disaster and to process this data, emphasizing Level-1 and Level-2 data fusion techniques. The result is an improved situation awareness that can be directed back into the simulation at (simulated) decision points in order to improve the management of the resources according to certain measures of performance and effectiveness. A scenario as complex as this one dictate that its simulation be

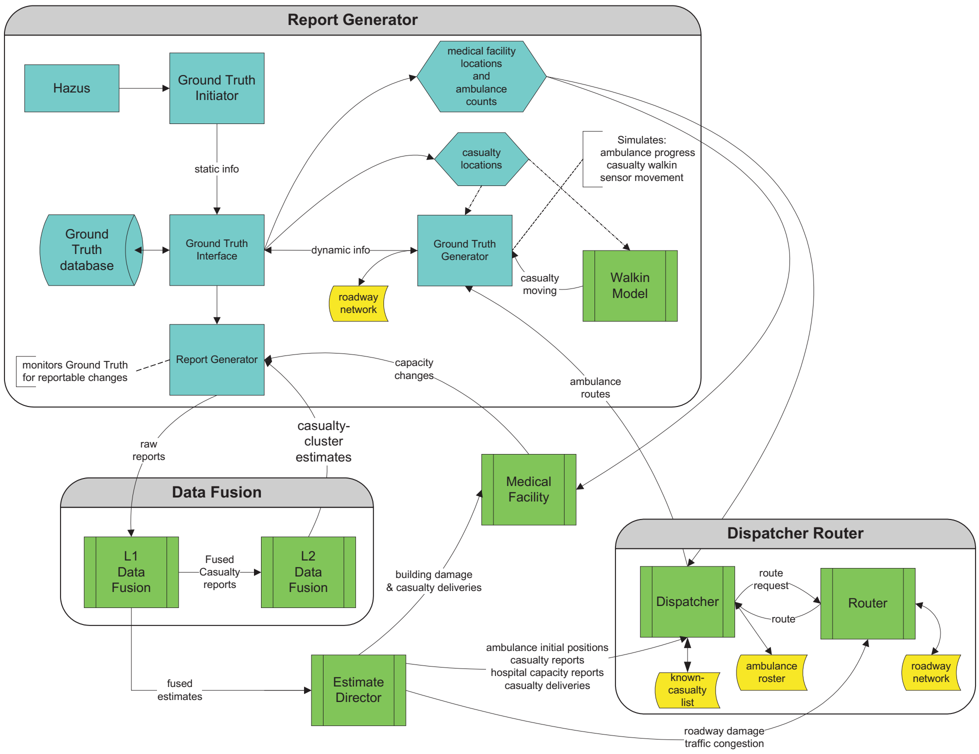


Fig. 1. Interfederate interactions

correspondingly complex. To that end the simulation model is implemented as a number of independent simulation modules (see figure 1).

2.1 Framework

The requirement for implementing several independent but cooperating models necessitates the use of a framework under which the various models can cooperate, each adhering to its own time-line and yet not encounter causality problems such as would be generated if each model ran to completion without some synchronization with the models on which it depends. Such a framework is the High-Level Architecture / Run-Time Infrastructure (HLA/RTI), developed under the leadership of the Defense Modeling and Simulation Office of the United States government. The HLA was developed to support reuse and interoperability across the large numbers of different types of simulations developed and maintained by the United States Department of Defense.

While the HLA is an architecture (not software), the use of RTI software is required to support operations of a multiple-model execution. The RTI software provides a set of services used by independent models to coordinate their operations and data exchange during a runtime execution (interested readers please refer to [1] and [2]). The entire agglomeration of models is called a *federation* and the individual models are called *federates* in this context. The sole communication among federates is a series of time-stamped messages, or *interactions*, which carry all inputs and outputs to and from each federate. A federate has three points of contact with the remainder of the federation- the message-receipt function, the message-sending function, and an *update* point which is called periodically and during which call, the state of the federate is updated to the next time increment value.

2.2 Dispatcher-Router's part in the simulation

The DR federate is a simulation of the two functions− dispatching ambulances and the calculation of the best route (to either a casualty or a medical treatment center). The task of dispatching and routing is broken into two independent decision systems. The dispatcher makes the decisions about which ambulance to send to a cluster or an individual casualty and then to which hospital. The router finds a path for each ambulance to complete its service, always capable of generating backup routes in case of a blockage due to the disaster. The dispatcher is also a decision point in the simulation where an improved estimate of casualty location and severity, derived from the information fusion module(s), is injected back into the simulation. Thus the simulation can be run either with or without the aid of data fusion, providing one rough measure of the effect of the availability of fused estimates. The dispatcher and the router are two parts of the same federate. The dispatcher dispatches EVs to a pickup location or a hospital depending on the number and type of casualties on-board. Router provides, when inquired, the quickest route from a source location to a destination while accounting for the effects of the disaster (such as damaged transportation infrastructure or geographic areas which must be avoided due to chemical or biological hazard).

The dispatcher receives reports of casualties (fused by a Level-1 fusion), and later, reports of casualty-clusters (developed by a Level-2 fusion) where the probability of finding severely injured casualties is high. The (simulated) dispatcher receives reports of ambulances waiting for dispatch, moving toward either a casualty or a hospital and even ambulances stopped at a previously unknown impassible obstacle (such as a collapsed bridge or tunnel) and awaiting either rerouting to their destination or a possible decision on dispatch to a new location. Figure 2 depicts the high-level information flow. Message receipt brings updates to the database from other simulation federates (e.g., casualty reports, casualty cluster reports, ambulance idle reports etc.). During the periodic updates, the dispatching strategy is implemented for all ambulances that currently have reached their destination (casualty pickup or hospital drop-off) and hence labeled *idle*. This is the trigger for ambulance routes to be generated and emitted as interactions to be sent to the model that generates the

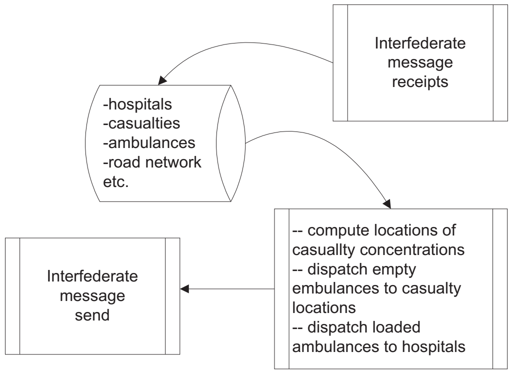


Fig. 2. High level data flow

simulated ambulance movement. The message receipts are asynchronous and can come at any time but the messages that are to be sent can be generated only during the update processing.

3 Literature review

In this section we now present the literature review explaining data fusion and the work that has been done regarding emergency response services, focusing mainly on dispatching and routing. To our knowledge, this kind of work has not been done in literature given such a huge scale of disaster and the real road network, although there is some related work that has been done on a smaller scale. Also use of information fusion to support the decision making of dispatching and routing is not considered in literature. The majority of existing Emergency Response Service (*ERS*) models deal with the following problems [3]:

(1) Location Problems: Location of ambulance stations within a region and satisfying requests for service subject to some performance criteria.
(2) Coverage Problem: Determination of the minimum number of ambulances to satisfy all calls for service.
(3) Other models dealing with
  (a) Various dispatching strategies and their influence on performance results.
  (b) Dispatching and routing using dynamic travel time information, re-deployment of resources and route changes under different system states.

3.1 Data Fusion

According to the definition given by the Joint Directors of Laboratories (*JDL*) Model-

> Data Fusion is a process dealing with the association, correlation, and combination of data and information from single and multiple sources to achieve refined position and identity estimates for observed entities, and to achieve complete and timely assessments of situations and threats, and their significance.

The process is characterized by continuous refinements of its estimates and assessments, and by evaluation of the need for additional sources, or modification of the process itself, to achieve improved results [4]. It involves the processing of information in the broadest sense to estimate the state of some aspect of the universe. Interested readers can also refer to [5] and [6] for more information on data fusion. The basic *JDL* Data Fusion Functional Model is briefly presented next:

Level 0 is the Sub-Object Data Association & Estimation. It involves pixel(signal) level

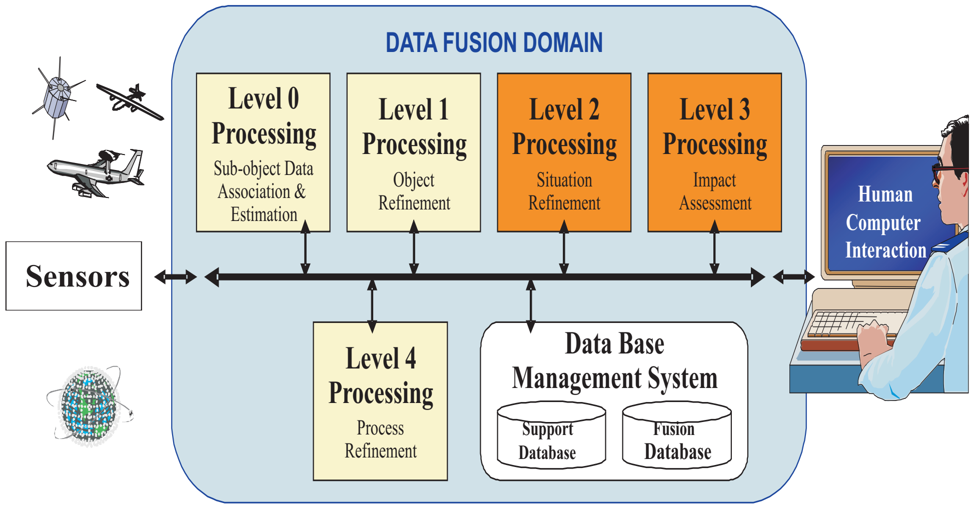


Fig. 3. Data fusion functional model

data association and characterization.

Level 1 is Object Refinement. It involves observation-to-track association, continuous state estimation (e.g. kinematics), and discrete state estimation (e.g. target type and Identification) and prediction.

Level 2 is Situation Refinement. It involves object clustering and relational analysis, to include force structure and cross force relations, communications, physical context, etc.

Level 3 is Impact Assessment(Threat Refinement). It involves threat intent estimation, event prediction, consequence prediction, susceptibility and vulnerability assessment.

Level 4 is Process Refinement. It involves adaptive search and processing (an element of resource management).

The paper by Rogova *et al* [7] describes the higher level fusion in context of a post-disaster situation. This paper reports recent progress on research focused on reusable and transferable situation assessment and impact prediction techniques to support crises management. It addresses the problem of higher level fusion to support the emergency operations and the purpose of situation assessment and impact prediction to provide decision makers with essential information to answer the questions often asked like - where are the problems, what kind of problems are there, and what is the impact of these problems. Higher level fusion processes comprise of contextual understanding of the characteristics of the state of the environment which are critical to decision makers, causal reasoning leading to recognition of possible causes of the identified events and estimated states of the world, and prediction of consequences of current situations. Out of the various examples of essential elements of information mentioned in [7], the DR federate is able to deliver significant inputs on:

(1) Regions of causalities (location, boundaries, severity of injury, etc.).

(2) Inaccessible or impeded transportation areas (due to road blockage or damage to the roads).
(3) Status of critical facilities, bridges etc.
Interested readers can also refer to the papers by Llinas [1] and Scott and Rogova [2] to get an overview of the fusion process in a post-disaster scenario under study. The casualty situation assessment and prediction given in [7] is shown in figure 4. The clustering algorithm used is also described in [7].

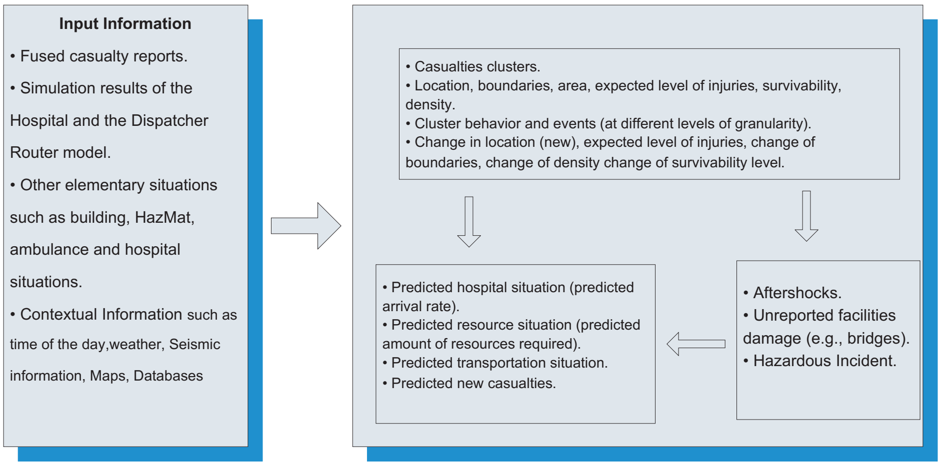


Fig. 4. Casualty Situation Estimation and Prediction

3.2 Emergency Response Services

There is limited literature that concentrates on routing of vehicles in a dynamic disaster environment, although Dynamic Vehicle Routing (DVR) problems have been studied in the past. Gendreau and Potvin [8] provide an overview of DVR problems. DRV problems refer to a wide range of problems where information of the problem is revealed to the decision maker concurrently with the determination of the solution. In these problems a set of vehicles is routed over a particular time horizon while new services are occurring in real-time. But none of this has been concerned directly with a dynamic disaster environment, where information is given as estimates by data fusion. The earliest papers in the literature on dynamic vehicle routing and dispatching were presented in the seventies and were either application-oriented or analytical [3]. Because real-time vehicle routing problems are NP hard and quick response times are required, exact algorithms are not yet capable of handling problems of realistic sizes. The core requirement for the development of an automated response system in such an environment is to have a fast heuristic that can provide good approximation results in the quickest possible time. For a large scale network, computation times to calculate shortest routes between a pair of origin-destination are prohibitively high. Zhan and Noon [9] gave a comprehensive performance comparison of 15 Shortest Path algorithms on a real road network and concluded that naive implementation of Dijkstra's algorithm should be avoided for large networks. Chou et al. [10] in their paper suggested an approach called *hierarchical approach* in which they proposed a hierarchical algorithm for approximating shortest paths between all pairs of nodes in a large scale network. Their algorithm begins by extracting a high level sub-network of relatively long links where routing decisions are most crucial. This high level network partitions the shorter links and their nodes into a set of lower level sub networks. By fixing gateways within the high level network for entering and exiting these subnetworks, a computational savings is achieved at the expense of optimality. We have used this approach in our case. This approach has been modified and used along with the support of data fusion.

Haghani et al. [3] developed a simulation model to evaluate a real-time EMS vehicle response system that uses real-time travel time information and assists the emergency vehicle dispatchers in assigning response vehicles and guiding those vehicles through non-congested routes. They used the transportation network, represented by 38 nodes and 62 links, of Arlington County, VA as the study network. Information obtained from analysis of the existing data was used to develop probability density functions describing the temporal, spatial and priority distribution of the request calls. They used historical travel time information and adjustment coefficients to get an average *off peak* and average *peak* travel time between any two nodes i and j. They also assumed that the historical average travel time will take care of all factors that affect travel time, such as incidents, congestions, and signal controls, etc. But existing records and historical data will not help in this case because of the dynamic nature of the post-disaster environment. Traffic flow was generated as a random variable with a normal distribution $N(\mu(t), \sigma(t)$, where $\mu(t)$ is the average flow rate at time t and is assumed available, and $\sigma(t)$ is proportional to $\mu(t)$. The travel speed on each link is determined using a unique speed flow relationship for that link.

This might not represent the real scenario which is dynamic in nature. In our simulation we are using the actual traffic flow and the actual travel speeds depending on the road condition and real time condition. We use a mix of strategies as described in the paper [3]$-$ first come first serve (FCFS), nearest origin assignment and flexible assignment$-$ for dispatching of ambulances as will be shown in later sections. Haghani et al. [11] developed an optimization model that, given real time traffic information, can assist emergency response vehicle dispatchers in assigning multiple emergency response vehicles to incidents and determining the routes for these vehicles that avoid hot congestion spots in the transportation networks. The model accounts for the service area coverage concerns (when several vehicles are busy) by relocation and redistribution of the remaining vehicles among stations. They use a dynamic shortest path algorithm given by Ziliaskopoulos and Mahmassani [12] to obtain the shortest paths between a given O-D pair. Sherali et al. [13] considered the problem of allocating certain available emergency response resources to mitigate risks that arise in the aftermath of a natural disaster, terrorist attack, or other unforeseen calamities. They derived a tight linear programming relaxation for the model formulation. But in there scenario there was no real time feed back on road conditions and casualty or cluster information and the situation is considered static. For a more comprehensive overview of this subject please refer to [14]. This paper is a review of the development and current state of the art in operations research for deployment and planning analysis pertaining to EMS and Fire Departments.

4 Data fusion

Lower level fusion is done for casualty reports and reports on roads (links). Based on all the reported information, the fusion center fuses the reports about the casualty and provides fused estimates about its injury class. Casualties are then classified into four injury types (classes) T_j $(j = 1, .., 4)$. Depending upon the reliability of the reporting sensors, a probability is assigned to each casualty regarding its injury class. Hence there are four different probabilities associated with the four injury types and each casualty has such a *injury-probability assignment* table associated with it. For example, the table for a casualty with ID 1007 (say), will look something like Table 1. Injury type of a casualty is given by

Table 1
Fused data about casualties injury type

Casualty ID	1007			
Injury Type	T_1	T_2	T_3	T_4
Probability	0.2	0.1	**0.4**	0.3

the injury type having maximum probability from its injury-probability assignment table which in this case is Type 3.

The information about the location of each patient is also imprecise. Instead of giving the exact spatial coordinates of each casualty, the fusion center estimates a *region of interest* around the casualty where it is likely to be present. The patient is located in the region with a probability no smaller than a given value $p(0 < p \leq 1)$. This is treated as a different problem that falls under the scope of *search theory.* Please refer to our forthcoming paper [15] for dealing with this search problem. In this particular simulation, the ambulances are dispatched to the nearest location of the region of interest and ambulance drivers pick up the casualty(s) they see first.

Similarly, for road conditions, the fused data gives us a level of damage associated with each link with a probability $(Pr(x))$ as shown in Table 2. This probability is the reliability of a particular source being accurate and is based on various factors. Some of the factors are− whether the source is an civilian, a police officer, some emergency response personnel or a firefighter, and whether there is any past information regarding the reliability and accuracy of the information given by a particular source. Hence each link has an expected level of damage, given by $\sum_i x_i L_i$ (%), and thus a corresponding traversal speed. The levels of damage considered in the simulation is also shown in Table 2. The percentage increase in the length corresponding to a level of damage is also shown in the table. At the start of every update interval, we update the network by increasing the corresponding link lengths. The routes are generated based on this updated network.

Table 2
Fused data about road conditions and percentage increase in length

Level of damage for each link	%age Damage	Probability of accuracy	%age increase in length
L_0	0%	$Pr(x_0) = 0.1$	0%
L_1	25%	$Pr(x_1) = 0.4$	25%
L_2	50%	$Pr(x_2) = 0.3$	50%
L_3	75%	$Pr(x_3) = 0.1$	100%
L_4	100%	$Pr(x_4) = 0.1$	Impassable

5 Dispatching

Our objective is to send the appropriate ambulances to casualties and then deliver those casualties to the appropriate hospitals. Thus, we consider this as two different problems: *patient pickup problem* and *patient delivery problem.* As the name suggests, *patient pickup problem* is concerned with the pick up of highest severity casualties with minimum service times. Here, service time is the time elapsed from the moment the casualty is discovered till the time it is picked up. *Patient delivery problem* is concerned with delivery of the picked up casualties to appropriate health centers for medical services while minimizing the service times. Service time in this case is given by the time elapsed from the moment the casualty is picked up and the time till it is delivered at the hospital. The pseudo-code is given in appendix D.

5.1 Patient pickup problem

In the patient pickup problem, an EV is dispatched (if idle) to a cluster of casualties or individual casualties. Preference is given to clusters since it is more efficient to process clusters than serve individual casualties because the ambulance is bound to find some casualty for pickup inside the cluster. If no clusters are present, the idle EV is sent to the nearest high severity casualty.

5.1.1 Method to choose a cluster

A cluster is reported when a number of casualties is detected in the same vicinity. Clusters are used as dispatch targets because the probability of finding casualties, enough to fill an ambulance, is high. Since a cluster can be of any size and shape, it is divided into a number of cells. This division can be based on census blocks in a census tract or any artificial sized blocks to make the cluster information usable. While dispatching to a cluster, the patient pickup problem involves two tier decisions− first determining which cluster should the ambulance be dispatched to and then determining which cell of that cluster should be serviced first. In order to choose an appropriate cluster, we need to consider three factors:
a) attractiveness of the cluster itself,
b) average distance from the ambulance to the cluster, and

c) availability of attractive hospitals around the cluster.
Clusters are chosen based on their respective attractiveness index calculated in the following manner:

1) *Attractiveness due to casualty type (CW_j)*
Attractiveness of a cluster depends on the number and type of casualties present in the cluster. Attractiveness of cluster j ($j = 1, ..., |J|$) is given by;

$$\alpha \sum_{i \in m_j} x_i + (1 - \alpha) \sum_{i \in m_j} y_i, \tag{1}$$

where,
m_j is the set of cells in cluster j,
x_i is the number of priority 3 casualties in cell i of cluster j,
α is the weight given to priority 3 casualties,
y_i is the number of priority 2 casualties in cell i of cluster j, and
$1 - \alpha$ is the weight given to priority 2 casualties.
Hence we chose the cluster that has the best attractiveness based on the casualty type and number. This is obtained by normalizing the attractiveness value of each cluster to obtain an attractiveness value (CW_j) for cluster j as:

$$CW_j = \frac{\alpha \sum_{i \in m_j} x_i + (1 - \alpha) \sum_{i \in m_j} y_i}{\sum_{j=1}^{|J|} \left[\alpha \sum_{i \in m_j} x_i + (1 - \alpha) \sum_{i \in m_j} y_i \right]}, \tag{2}$$

where $|J|$ = total number of clusters. The cluster $i = argmax\ CW_j$ is selected for dispatch.

2) *Attractiveness due to distance (D_j)*
A simple method to calculate the distance from the current location of an ambulance to a cluster is to consider the gravity center of the cluster and calculate the distance based on that. But this method is not appropriate for our scenario since a cluster can have any size and shape. The drawback is explained through an example. Consider a cluster with a *'donut'* shape. The gravity point is located at the center of the donut. However, there are no cells in the vicinity of the center, and hence not suitable for dispatch. We employ a method which takes the average of the distances between all the centers of the cells and the location of the ambulance. We then normalize the attractiveness index due to distance, D_j, given by the expression:

$$D_j = \frac{d_j}{\sum_{j=1}^{|J|} d_j}. \tag{3}$$

Here, $d_j = \left\{ \frac{\sum_{k \in m_j} SLD_k}{|m_j|} \right\}$ is the effective distance between the ambulance and cluster j.

SLD_k is the straight line distance from ambulance to cell k of cluster j,
$|m_j|$ is the total number of cells in cluster j, and
$|J|$ is the number of clusters.
The cluster with the least value of D_j is the best candidate for dispatch based on distance criteria.

3) *Attractiveness due to hospitals (HW_{jh})*
While choosing a cluster we also need to take into account the surrounding hospitals. It is better that we dispatch to a cluster that is close to the hospitals to facilitate the delivery of casualties to one of them. Hence, similar to the calculation of D_j in equation 3, the average distance from cluster j to all hospitals is normalized to get:

$$DH_{jh} = \frac{\sum_{h} \sum_{i \in m_j} SLDH_{ijh}}{|m_j|}.$$

The attractiveness of cluster j to hospital h is given by:

$$HW_{jh} = \frac{DH_{jh}}{\sum_{j=1}^{|J|} DH_{jh}}. \tag{4}$$

Here, $SLDH_{ijh}$ is the straight line distance from cell i of cluster j to hospital h. Hence based on the attractiveness index due to the above three factors, cluster j^* is chosen for service by the ambulance and patients are transported to hospital h^*, where

$$(j^*, h^*) = arg\ max_{j,h}\{W_1 CW_j - W_2 D_j + (1 - W_1 - W_2) HW_{jh}\}, \tag{5}$$

where,
W_1 = weight associated with a cluster,
W_2 = weight associated with distance between the ambulance and cluster j, and
$1 - W_1 - W_2$ = weight associated with the attractiveness of hospitals.

5.1.2 Method to chose a cell

The next (lower level) decision is to determine which cell i^* of the cluster j^* should be served first so that we are guaranteed of finding the casualties in the shortest possible time. The attractiveness of a cell to an ambulance is based on the attractiveness of the cell itself (given by the number and type of casualties inside it), the average attractiveness of its neighboring cells, and the distance from the ambulance to the cell.

1) *Attractiveness of an individual cell i due to casualty type (CW_i^j)*
The normalized attractiveness value of a cell due to the number and type of casualties is determined in the same way as in equation 2. Thus, we have

$$CW_i^j = \frac{\alpha x_i + (1-\alpha) y_i}{\alpha \sum\limits_{i \in m_j} x_i + (1-\alpha) \sum\limits_{i \in m_j} y_i} . \tag{6}$$

2) *Attractiveness due to neighbors of cell i in cluster j (NW_i^j)*
Due to the uncertainty of the data, it might happen that the ambulance does not find casualties in a designated cell. In order to save time and avoid re-dispatching, the ambulance searches the vicinity of that cell to find casualties. Hence, it is always better to dispatch ambulances to the cell that has attractive neighbors. Using equation 6 the attractiveness value due to neighbors of cell i is given by

$$NW_i^j = \frac{\sum\limits_{k \in n_i} CW_k^j}{|n_i|} , \tag{7}$$

where n_i is the set of neighbors to cell i. The neighbors of a cell are the cells that share at least one common boundary (out of four) with it.

3) *Attractiveness due to distance (D_{ia})*
Finally, the attractiveness due to the distance between an ambulance a and cell i is given by

$$D_{ia} = \frac{SLD_{ij}}{\sum\limits_{i \in m_j} SLD_{ia}} , \; where \tag{8}$$

D_{ia} is the normalized distance value between cell i and ambulance a and SLD_{ia} is the straight line distance between ambulance a and cell i.
The cell to which the ambulance is dispatched for casualty pickup in cluster j is determined by the expression

$$i^* = arg\ max_i \left\{ W_1^j CW_i^j - W_2^j D_{ia} + (1 - W_1^j - W_2^j) NW_i^j \right\} . \tag{9}$$

Where;
W_1^j = weight associated with a cell i in cluster j,
W_2^j= weight associated with the distance between the ambulance and cell i in cluster j, and
$1 - W_1^j - W_2^j$ = weight associated with average value for neighbors of cell i in cluster j.

5.2 Patient delivery problem

Once casualties have been picked up, the next task is to deliver them to appropriate hospitals. to decide the appropriate hospital we consider the hospital's available capacity, the waiting time for each priority type, and the distance from the pickup location. For a casualty picked up in cluster j, the hospital for delivery, h^*, is given by:

$$h^* = arg\ max_h \{\delta_c CAP_h - \delta_d DEL_h - (1 - \delta_c - \delta_d) dh_{jh}\} \ , \tag{10}$$

where,
δ_c = weight associated with capacity of a hospital,
δ_d = weight associated with delay of a hospital,
$1 - \delta_c - \delta_d$ = weight associated with average distance between a cluster and hospitals.

The average capacity in hospital h based on the number of type 2 and type 3 casualties the hospital can take in for treatment is given by:

$$CAP_h = \frac{\beta(CAP_3)_h + (1-\beta)(CAP_2)_h}{\sum_{h=1}^{|H|} \beta(CAP_3)_h + (1-\beta)(CAP_2)_h}, \tag{11}$$

where,
$|H|$ = total number of hospitals,
β and $(1 - \beta)$ = weights associated with capacities for priority 3 and priority 2 casualties respectively,
$(CAP_i)_h$ = capacity of priority i $(i = 2, 3)$ casualties in hospital h.

The average delay in hospital h based on the waiting times for both the type 2 and type 3 casualties is given by:

$$DEL_h = \frac{\gamma(DEL_3)_h + (1-\gamma)(DEL_2)_h}{\sum_{h=1}^{|H|} \gamma(DEL_3)_h + (1-\gamma)(DEL_2)_h}, \tag{12}$$

where,
γ and $(1 - \gamma)$ = weights associated with delays for priority 3 and priority 2 casualties respectively,
$(DEL_i)_h$ = waiting time of priority i $(i = 2, 3)$ casualties in hospital h.

The normalized distance of cells in cluster j to hospital h, dh_{jh}, is given by:

$$dh_{jh} = \frac{d_{jh}}{\sum_{j=1}^{|J|} d_{jh}}, \tag{13}$$

where $d_{jh} = \frac{\sum_{i \in m_j} SLD_{ih}}{|m_j|}$, with SLD_{ih} being the straight line distance from cell i of cluster j to hospital h.

6 Routing

On large scale networks like the we are considering, with tens of thousand of nodes and links, computation times are prohibitively high when we apply the traditional Dijkstra's shortest path algorithm (c.f.[9]). In addition, actual link travel times are determined by traffic congestion and road conditions. Hence the data (network) used in computing the shortest paths has to be updated periodically. Also during each simulation update interval, we get a large number of requests for routes. To overcome this computational problem, we use a hierarchical approach detailed in section 6.3. Since in a disaster situation, any node can be a potential origin or a destination, a fast heuristic that can provide good approximation results in a limited amount of time is preferred to exact methods.

6.1 Link status updates

One of the issues which have to be addressed in the routing process includes updating the link status. Initially the link status of the whole network is assumed to be at some particular state based on past data about the traffic conditions (depending on the time of the day when the simulated earthquake strikes). Post-disaster fused data estimates about each link are given by the data fusion center. For each link that is reported to be damaged, we update the link length according to the procedure suggested in section 4 (Table 2).

6.2 Congestion

Another issue is the congestion on links, which has a effect of increasing the travel time. By congestion we mean that the road segment(s) can not be traversed at normal speed due to various restrictions that make the traversal slower. Congestion might be due to any of the following reasons:
1. All other roads might be damaged and it is the only way to go to a particular destination.
2. The traffic density is very high on that particular road.
3. Traffic from other regions is being diverted to this road.
4. Congestion due to road damage (with usual traffic).
The effect of congestion is reduced speed and hence more travel time. This effect can be taken into account by increasing the link lengths in the network data base by a factor

corresponding to the reduction in speed due to the congestion. We get reports on traffic congestion and these reports are fused by L_1 fusion and the fused estimates about each link are then sent to the DR federate. Table 3 shows the delay due to congestion and the corresponding increase in link lengths. Delays are reported as a number between 0 to 100 and its interpretation is given in the table.

Table 3
Delay due to congestion and percentage increase in length

Delay on each link	%age Damage	%age increase in length
$0 - 20$	$\leq 20\%$	0%
$20 - 40$	$\leq 40\%$	25%
$40 - 60$	$\leq 60\%$	50%
$60 - 80$	$\leq 80\%$	100%
$80 - 100$	$\leq 100\%$	Impassable

6.3 Routing Methodology

The Hierarchical Approach (HA) suggested in [10] is used to partition the network into smaller manageable networks to find the shortest paths between *O-D* pairs. We use Dijkstra's Shortest Path Algorithm to find the shortest path between a given *O-D* Pair. As in [10] two levels of hierarchy are defined - *Upper Level* and *Lower Level*. Upper level contains links that are major or minor interstate and US state highways. It is called the *Macro Network*. Lower level contains the sub-networks that are formed by the enclosures of the first level network. These are called the *Micro Networks*. *Gateway Nodes* are marked for each micro-network. Gateway nodes are the nodes where from we can enter or exit a particular micro-network. These are marked on the boundary for each micro-network, i.e. they are chosen from the macro-nodes(nodes belonging to macro network). The authors of [10] define macro-network as the high-level network, reflecting the fact that it gives a macroscopic view of the original network. Correspondingly, they call the low-level subnetworks as micro-networks, reflecting the fact that it gives a microscopic view of a part of the original network.

We used ArcGIS software developed by ESRI (interested readers please refer to the web site *www.esri.com*) for the decomposition of the network into the macro-network and its micro-networks, and for choosing the gateway-nodes used for entry (or exit) to (or from) a micro-network. The procedure is shown briefly in the case study. The main idea of our approach is same as in [10] but some modifications are done for use in our scenario, as will be depicted later in the case study. Instead of choosing only the highways as the macro-arcs, we upgrade some micro-arcs to macro-arcs so as to better define the micro-networks making them suitable for application of Dijkstra's algorithm. The authors of this paper suggest to choose the actual entry/exit ramps for a major highway as the only gateway nodes. But, instead, we choose 6-10 gateway-nodes, not necessarily the actual exit ramps on a major highway, for each micro-network depending on the size of the micro-network. This is done since we do not want to fix only one gateway-node for entry (or exit) to the macro-network and get stuck in the routing procedure in case the entry (or exit) point gets damaged or blocked due to the earthquake. We modify the *best node* approach given in [10], which chooses the best gateway-node to enter/exit any micro-network as described next.

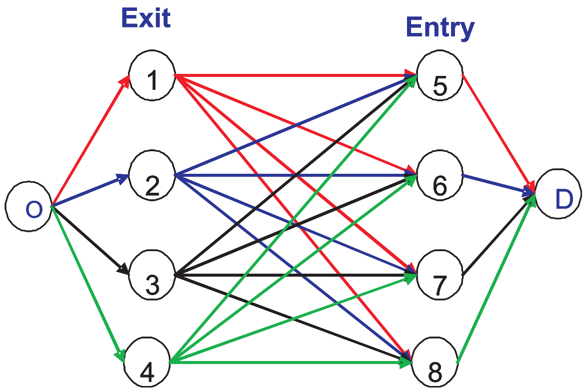


Fig. 5. Choosing the Best Route

Refer to figure 5. Find the shortest path from origin O to the exit points n_i of region i (in this case $n_i = 4$). Next, find the shortest path from each exit point n_i of region i to every entry point m_j of region j (in this case $m_j = 4$). Finally, find the shortest paths from each of the entry points m_j of region j to the destination point D. In all we will get $n_i \times m_j$ different paths from O to D and we choose the best out of these paths. We will call this the *Best Exit-Entry* approach. We do not use the *nearest node* approach suggested in [10] since the aim here is to find the path quickly rather than the actual shortest path. Finding the final path for an *O-D* pair will vary depending on which micro-network the origin/destination fall in. Four cases are possible:

1) Both the origin and destination are in the same micro-network or on a macro-network. Hence there are no entry/exit points and we will simply apply Dijkstra's algorithm to find the shortest path.

2) If they are in different micro-networks then find the shortest path as described above.

3) If the origin is on the macro-network and destination is inside a micro-network, the procedure remains the same except that now there is only one exit point (same as origin) and hence we have only m_j paths to chose from.

4) If the destination is on the macro-network and origin is inside a micro-network, the procedure remains the same except that now there is only one entry point (same as destination) and hence we have only n_i paths to chose from.

7 Case study

In this section we will first present some of the assumptions and guidelines we used for dispatching purposes and then the format for the network of the study area. The difficulties in conversion to a suitable format for use in our simulation are cited and the modifications done to rectify these problems are discussed. We also discuss the application of the approach of [10] to our scenario. and the modifications needed to that approach. Finally we present some representative computational results.

7.1 Assumptions, dependencies, constraints and guidelines

The goal of the DR federate is to develop an awareness of the locations of (simulated) human casualties and to dispatch ambulances to those casualties (and thereafter to a hospital or a treatment center). Interactions for the DR federate are given in appendix A. We now specify some of the assumptions, dependencies and constraints:

(1) The area chosen for study is the Los Angeles basin with the earthquake simulating rather closely the Northridge event of 1994.
(2) The Ground Truth, from which much of the simulated phenomenology is derived, has been generated by the HAZUS software and includes both human casualties and building damage.
(3) The simulation is not intended to represent activity beyond 24 hours post-event (typically much less time).
(4) The simulation is a time-based model that is updated on a regular basis.
(5) Receipt of an *Ambulance Idle* message will indicate that that ambulance has no destination and is a candidate for dispatch.
(6) Receipt of an *Ambulance Idle* message with a casualty count of 0 will indicate a dispatch of that ambulance to a location that has a high probability of containing at least 3 casualties.
(7) Receipt of an *Ambulance Idle* message with a casualty count of 1 or 2 will indicate a dispatch of that ambulance to a location that contains casualties. The choice of destination will depend on the number and severity of casualties already on-board.
(8) Receipt of an *Ambulance Idle* message with a casualty count of 3 will indicate a dispatch of that ambulance to a hospital that has a high probability of having available capacity to treat the casualties.
(9) If on the occasion of a dispatch to a hospital, even if no hospital has any available capacity, the hospital with the best result from the attractiveness computation will still receive the dispatch.
(10) Dispatcher Router will receive reports of casualty clusters. Such reports will present a cluster as a list of cells, each containing a count of reported casualties of both severity 2 and severity 3.
(11) There is no persistence in the reporting of clusters. Thus a cluster is considered to exist only from the time that it is reported until the next time the clusters are reported. Subsequent cluster reports will be considered to refer to independently-defined clusters. It is assumed that whenever higher level fusion reports clusters, it considers all the present and past information and gives estimates about the most recent state of the scenario.
(12) Casualties will be picked up by an ambulance at a dispatch location from the casualties closest to the destination node without regard for cell membership in case of a dispatch

to a cluster(which the simulated ambulance driver knows nothing about).

(13) In case of dispatch to a casualty, an ambulance will pick up a casualty regardless of its ID from the casualties closest to the destination node specified to it.
(14) Ambulances can travel a link in any direction.
(15) The maximum capacity of each ambulance is 3.
(16) The ambulance will always try to pickup the maximum number of casualties.
(17) Type 4 casualties are mortally injured and hence not considered for pickup and delivery for obvious reasons of facilitating the service of those casualties that can still be saved.
(18) Type 1 casualties are the ones that do not need immediate medical attention and hence can be ignored.
(19) Data about road damages, traffic conditions and congestion on the roads is already available (from data fusion center). This data is summed up as a *delay factor* for each link.
(20) The capacities of the disaster area hospitals are reported as the numbers of injuries that can be treated in a given time window.
(21) There exists a model of disaster area hospitals that can provide such capacity estimates. Interested readers please refer to the paper by Yi et. al [16].
(22) The condition of a patient may deteriorate while he/she is waiting in queue and hence such casualties have to be upgraded to higher casualty types. Hence a casualty can be reported more than once.

7.2 Network details

The Northridge region in the county of Los Angeles is chosen as the test network. The network has 34,890 nodes and 43,445 links. The input network was given in the form of a Tele-Atlas database file (appendix B contains information regarding the data format). In addition to this, each link is associated with a specific *road type*. Various road types tell us the speed with which they can be traversed. The road types are:

(1) Class 0, 1 and 2: Interstate highways (traversal speed $55 - 65$ mph).
(2) Class 3: High-speed roads, major arteries and ramps (speed $\leq 40 mph$).
(3) Class 4, 5 and 6: Small roads, slow speed ramps etc. (speed $25 - 30$ mph).
(4) Footpaths, shore lines etc. (not suitable for traversal).

7.3 Problems and rectifications

To make use of the Tele-Atlas file, it was first converted to a *shape-file* for use in ArcGIS. ArcGIS facilitates working with the network since everything can be seen and done visually. A hierarchical approach is used within ArcGIS to decompose the network. It stores the information in a **.dbf* file which is much like an *Microsoft Excel* file and contains the information about the network− arc lengths, node ID's, link ID's etc. The given network converted from Tele-Atlas database file is shown in figure 6. A sample attribute table(*.dbf file) is presented in appendix C that gives the type of data extracted from the Tele-Atlas file. Each link has a link ID and/or a name associated with it. Other attributes associated

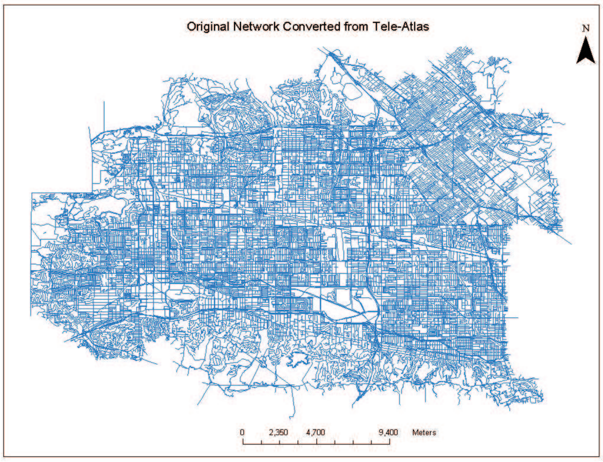
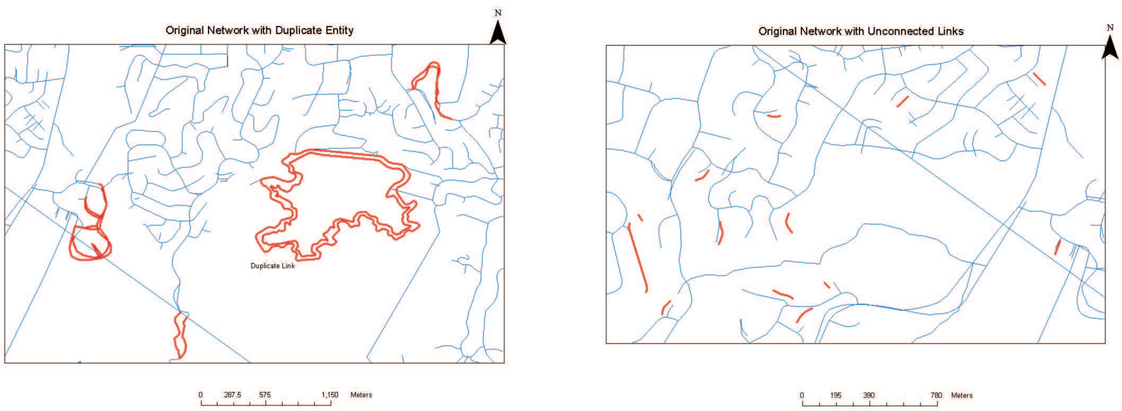


Fig. 6. Original network

with a link are length, its two end nodes (*from ID & to ID*), and damage or delay to the link. It also specifies if a link is a bridge or a tunnel. In addition to this, each node has a X-Y coordinate (UTM co-ordinate) associated with it.

As is usual in any data transformation, the text file from Tele-Atlas when converted to a shape-file, had many errors in it. The new shape-file had many links with no IDs. There where many links which were not connected to any other link and many of them were duplicate links (refer figure 7(a) and figure 7(b)). All other links like footpaths and shorelines are unnecessary in the network and hence should be deleted. No shortest path algorithm can



(a) Duplicate

(b) Unconnected

Fig. 7. Original network with duplicate entities and Uuconnected links

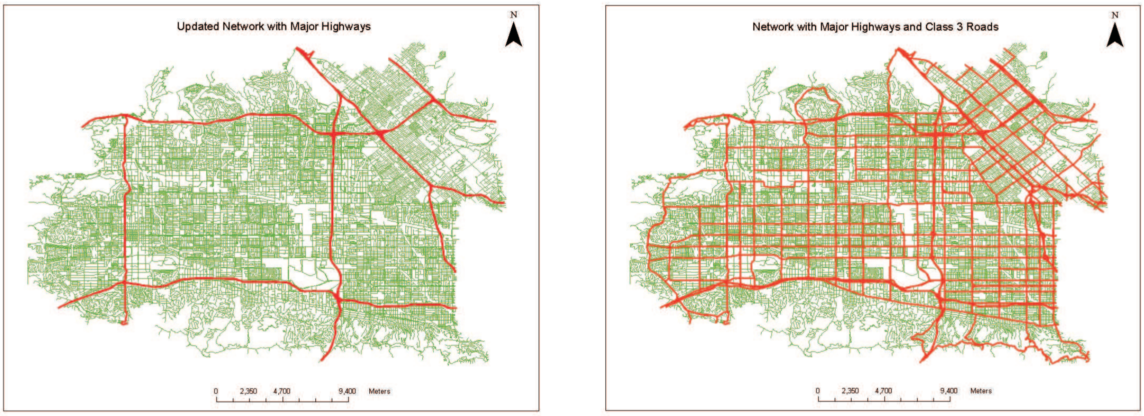
be applied on such a network. Moreover the time for calculations might also increase due to these errors in the network. Hence some rectifications of the network are necessary and these modifications are done manually in ArcGIS. The modifications done were:

(1) Unconnected arcs were found and deleted since there is no way to reach these disconnected links.

(2) Duplicate links were found and deleted since duplicate links will make the calculation of shortest paths slower.
(3) Links with no IDs (0 ID links) were checked for and deleted if
  (a) They are duplicates of some other links.
  (b) They are unconnected.
  (c) They are dangling links. Most of these dangling links are connectors to some other landmarks and might not be important and hence are deleted.
(4) All other 0 ID nodes were assigned ID's so that they can be used in the network.
(5) Links like footpaths, shorelines, and political boundaries etc. were deleted since they do not have any meaning for routing.

7.4 Hierarchical approach

By applying the hierarchical approach on the above (rectified) network, the network we got is shown in figure 8(a). As per the definition of macro-arcs (the arcs that constitute

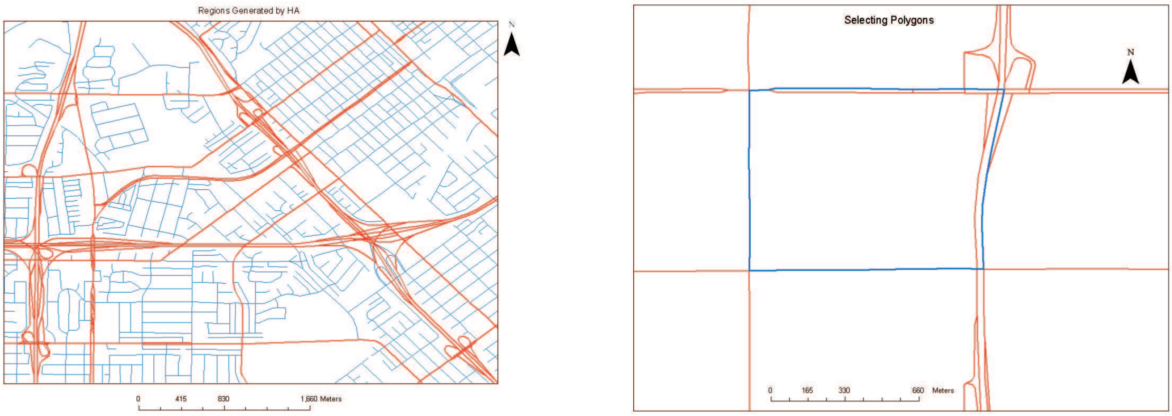


(a) Network obtained using major highways only

(b) Updating class 3 links to macro-links

Fig. 8. Updated network using hierarchical approach

the macro-network) suggested in [10], we included class 0, 1 and 2 roads into the macro-network. As we can see the micro-networks, thus generated, are not of much help for finding shortest paths quickly as the number of links in each micro-network is still large and do not serve our purpose of decomposition. Hence we upgraded class 3 roads as macro-links resulting into the network as shown in figure 8(b). This does not serve the purpose either, as it breaks up the whole network into too many micro-networks. Too many lower level arcs in the macro-network will mean that the travel is slower on these segments and hence the route the ambulance takes is not efficient. The number of regions (micro-networks defined by the bounding box created by the arcs of the macro-network) created is large and will be difficult to manage and also increase the computation times. Moreover, as can been seen in figure 9(a), there would be hundreds of regions with just a few or no links inside it(by definition any region bounded by the macro-arcs is a micro-region). Also there will be regions inside regions, and regions intersecting regions. This gives us no proper way to use the network hierarchy



(a) Regions inside regions (b) Selecting polygons

Fig. 9. Creating polygons to form separate regions

and the suggested approach for finding shortest paths. We need regions that will be adjacent and comparable in size to each other. The micro-network should neither be too sparse nor too dense. So we create regions of proper size and each region contains approximately 400-700 links in it. This is done in ArcGIS by selecting the regions manually. To have individual regions we start by selecting various arcs from the macro-network one at a time, such that a polygon is formed and all the polygons share at least one side with another polygon thus forming a grid like structure(figure 9(b)). Wherever necessary (outer micro-networks), we

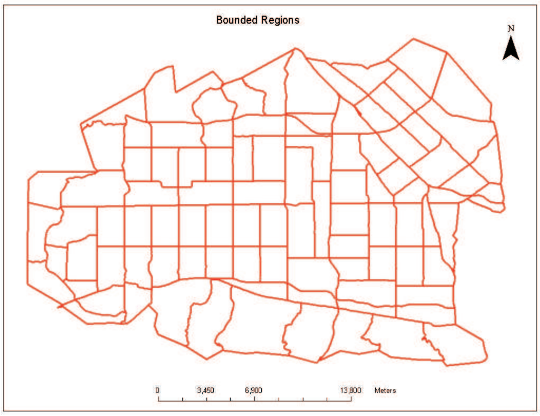
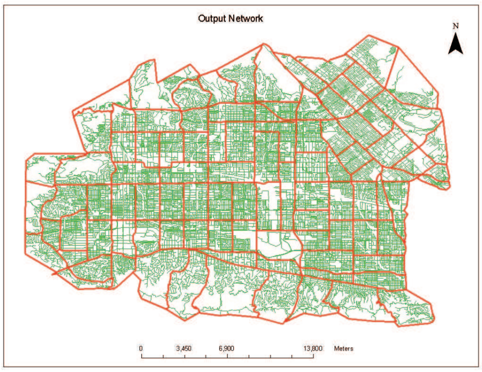


Fig. 10. Network with all the regions

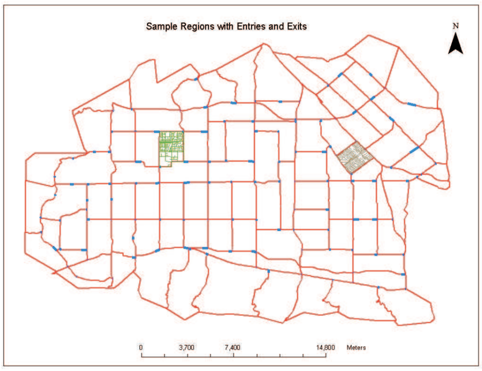
use pseudo-links so as to create bounding boxes that defines the regions correctly. These pseudo links are not used in calculations of a shortest path. The output network with all the regions is shown in figure 10.

7.5 Network partition

To actually create partitions and decompose the network into a macro-network and its related micro-networks, we select a polygon defining the boundary of each region individually and then select those links that fall within this polygon. We do this for all regions and the network is partitioned into 98 regions. We have 99 shape files− one for the macro-network and 98 for the micro-networks (refer figure 11(a)). Each region in itself is a network and we apply shortest path algorithm to find the shortest paths within each region by applying the *best exit-entry* approach described in section 6.3. After creating the micro-network we now have to decide on the entry-exit points that will be the gateway-nodes for each region. This is done keeping in view the shape of the region, its area and the adjacent regions and their entry-exit points. This is an important selection, as this will decide as to where from we get out of a particular region and enter a new one. This has an effect of increasing or decreasing the length of the tour an EV has to travel. The selected entry/exit (gateway nodes) along with a few regions is shown in figure 11(b). This final partitioned network is used to get a



(a) Final partitioned network



(b) Network with exit/entry (gateway) nodes

Fig. 11. Final network : Input to dispatcher-router

shortest path between an origin-destination pair. We determine which region(s) an origin and the destination belongs to and its corresponding exit/entry nodes. Then according to the *best exit-entry* approach, we find the shortest path.

7.6 Results

We used data to simulate one hour of the post-disaster scenario. There are approximately 76 ambulances in the Northridge region and 20 hospitals. We assume that there are 15 emergency phone lines at the ERC and a casualty is reported every 30 seconds. The simulation of all the other federates is not complete yet, hence we can not show the actual results of the whole simulation with all the federates running in synchronization with each other. To that effect, we have created a test driver that generates the reports as would be generated by Report Generator (ground truth). The reports that have been created by the test driver are− reports on casualty pickup, reports on casualty delivery and reports on stuck ambulances. These reports are generated keeping in consideration the actual travel times the ambulance would have taken to reach a destination. During the simulation a total of 1190 casualties were generated and a total of 37 clusters were reported. The delay due to congestion is generated by HAZUS. HAZUS generated an average damage equal to 2.5 (i.e. approximately 65% damage) and an average delay equal to 40 (i.e. approximately 40% delay on links). A total of 34 ambulances were reported as being stuck during the simulation period.

To investigate the effect of various parameters used in the dispatching methodology in section 5 we ran the simulation using seven different sets of parameters. Table 4 shows the simulation setup using these seven parameter settings. The first setup is the recommended one . The remaining setups change some of the parameters to measure their effect on the corresponding dispatching decisions. In the second and third setups, the weights associated with dispatch to a cluster are changed keeping all other parameters fixed. In the fourth and fifth setups, weights associated with dispatch to a hospital are changed and for the last two setups we use a different mix of parameters. The results are shown in table 5. Response time in case of a casualty pickup is the time at which the casualty is picked up minus the time the casualty is first discovered/reported. Similarly, the response time for a cluster is the time at which we start to serve the cluster minus the time it was first discovered/reported. Response time in case of a casualty delivery is the time at which the ambulance is reported as idle (after it has picked up the casualty) minus the time it reaches a hospital. The table presents the average distance the ambulance has to travel in order to go to a casualty pickup location, a cluster cell or to a hospital location for service. It shows the total number and type of casualties served depending on the casualty parameter (a smaller value of the parameter implies a greater preference of casualty type over distance). Running time (in clock minutes) of the simulation, on a HP Workstation with Pentium 4 processor and 520 MB of RAM, is also given in the table. Figure 12(a) plots the average response time to a casualty measured from

Table 4
Setup for various parameters used in the simulation

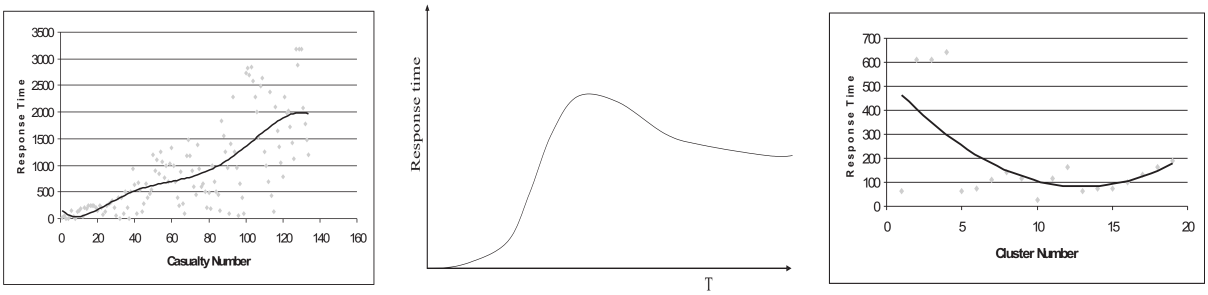
Parameter	**Setup Number**						
	1	2	3	4	5	6	7
Cluster Parameters							
Type 3 Weight (α)	0.6	0.6	0.6	0.6	0.6	0.7	0.5
Type 2 Weight ($1-\alpha$)	0.4	0.4	0.4	0.4	0.4	0.3	0.5
Cluster Weight(W_1)	0.5	0.7	0.5	0.5	0.5	0.7	0.6
Distance Weight (W_2)	0.2	0.3	0.2	0.2	0.2	0.3	0.1
Hospital Weight ($1-W_1-W_2$)	0.3	0.0	0.3	0.3	0.3	0.0	0.3
Individual Cell Weight (W_1^j)	0.4	0.4	0.5	0.4	0.4	0.5	0.4
Distance Weight (W_2^j)	0.3	0.3	0.1	0.3	0.3	0.1	0.2
Neighbor Cells Weight ($1-W_1^j-W_2^j$)	0.3	0.3	0.4	0.3	0.3	0.4	0.4
Hospital Parameters							
Capacity Weight (δ_c)	0.4	0.4	0.4	0.6	0.4	0.6	0.5
Delay Weight (δ_d)	0.3	0.3	0.3	0.3	0.6	0.3	0.3
Distance Weight($1-\delta_c-\delta_d$)	0.3	0.3	0.3	0.1	0.0	0.1	0.2
Type 3 Weight(β,γ)	0.6	0.6	0.6	0.6	0.6	0.7	0.5
Type 2 Weight ($1-\beta$,$1-\gamma$)	0.4	0.4	0.4	0.4	0.4	0.3	0.5
Casualty Parameters							
Type 3 Weight	0.7	0.7	0.7	0.7	0.7	0.5	0.3

the point when the casualty is first discovered/reported and the time an emergency vehicle is dispatched to it. We see that, initially the average response time is small since it is the start of the simulation and we have many idle ambulances, and then it starts increasing quickly

Table 5
Result comparisons for all setups

Field	Setup Number						
	1	2	3	4	5	6	7
Avg Response Time to Cluster (Min.)	10.06	9.7	10.06	10.06	10.06	10.26	9.89
Avg Distance to Cluster (Miles)	6.3	6.4	6.3	6.3	6.3	6.5	6.2
Avg Response Time to Casualty (Min.)	22.2	22.2	22.2	22.2	22.2	22.3	22.3
Avg Distance Casualty (Miles)	5.5	5.5	5.5	5.5	5.5	5.5	5.5
Total Type-3 Served	105	105	105	105	105	110	122
Total Type-2 Served	29	29	29	29	29	24	12
Avg Response Time to Delivery (Min.)	8.3	8.3	8.3	8.9	8.6	9.3	8.6
Avg Distance Delivery (Miles)	6.1	6.1	6.1	6.1	6.4	7.0	6.3
Running Time (Min.)	31	27	30	32	30	28	31

as the ambulances get busy. The actual behavior of the average response time is shown in figure 12(b). Response time increases up to a certain point and then starts to stabilize after a while. Figure 12(c) shows the average response time to a cluster measured from the point when it is first discovered/reported and the time an ambulance is dispatched to it. Clusters start appearing around 20 minutes after the simulation has started. Initially the response time is more since the ambulances are busy serving the casualties. After a while, when ambulances start becoming idle, the response time decreases. This is because we give dispatching preference to clusters over individual casualties that are not part of a cluster. It then starts increasing again as ambulances get busy. Eventually the response time should stabilize (if we let the simulation run long enough).



(a) Average response time for casualties (b) Average response time behavior (c) Average response time for clusters

Fig. 12. Average response time plots

8 Conclusions and future work

This paper investigated various strategies for dispatching and routing of emergency vehicles in a dynamic disaster environment. For routing decisions our suggestion is to suitably partition the transportation network into manageable regions (the process of doing this is illustrated in detail). For dispatching decisions, our suggestion is to incorporate both hospital delays as well as travel delays on links into the decision making process. Experimentation on parameter settings is needed to get good results and such an experiment is illustrated on a sample case study of an earthquake scenario in Northridge, California.

Future work can include the modeling of chemical spills and fire breakouts and their impact on dispatching/routing decisions for emergency vehicles. Also, the effect of migrating population (those who are not injured or have minor injuries) on traffic congestion needs to be modeled. Finally, a detailed analysis of casualty search times would improve the fidelity of the model.

References

[1] Llinas J. Information fusion for natural and man-made disasters. *In Proceedings of the 5th International Conference on Information Fusion, Annapolis, MD*, pages 570–574, July 2002.

[2] Scott P. and Rogova G. Crisis management in a data fusion synthetic task environment. *In Proceedings of the 7th International Conference on Information Fusion, Stockholm, Sweden*, pages 330–337, 2004.

[3] Haghani A., Tian Q., and Hu H. A Simulation model for real-time emergency vehicle dispatching and routing. *Transportation Research Record, No. 1882*, pages 176–183, 2004.

[4] Franklin E. White. Data fusion lexicon, Joint Directors of Laboratories. *Technical Panel for C3, Data Fusion Sub-Panel, Naval Ocean Systems Center, San Diego*, 1987.

[5] CMIF. Center for Multisource Information Fusion, University at Buffalo. *url: http://www.infofusion.buffalo.edu/tutorial.htm.*

[6] Steinberg A.N., Bowman C.L., and White E.F. Revisions to the JDL data fusion model. *Proceedings of the SPIE Sensor Fusion: Architectures, Algorithms, and Applications III*, pages 430–441, 1999.

[7] Rogova G., Scott P., and Lollett C. Situation assessment for the post-disaster casualty mitigation operations. *The 8th International Conference on Information Fusion, Philadelphia*, 2005.

[8] Gendreau M. and Potvin J. Y. Dynamic vehicle routing and dispatching. *In Fleet management and logistic. Kluwer Academic Publishers. T.G. Crainic, G. Laporte eds.*, pages 115–226, 1998.

[9] Zhan F.B. and Noon C. E. Shortest path algorithms: An evaluation using real road networks. *Transportation Science, Vol.32, No.1*, pages 65–73, 1998.

[10] Chou Y.L., Romeijn H.E., and Smith R.L. Approximating shortest paths in large scale networks with an application to intelligent transportation systems. *INFORMS Journal on Computing, Vol. 10*, pages 163–179, 1998.

[11] Haghani A., Siani Y., and Hamedi M. An on-line vehicle dispatching and routing model with area coverage constraints. *Proceedings of the 85th Annual Meeting of the Transportation Research Board, Washington D.C.*, Jan, 2006.

[12] Ziliaskopoulos A. and Mahmassani H. Time dependent, shortest path algorithm for real-time intelligent vehicle highway system applications. *Transportation Research Record 1408*, pages 94–100, 1993.

[13] Sherali H.D., Desai J., and Glickman T.S. Allocating emergency response resources to minimize risk with equity considerations. *American Journal of Mathematical and Management Sciences, Vol.24 , Nos. 3 and 4*, pages 367–410, 2004.

[14] Goldberg J. B. Operations Research models for the deployment of emergency services vehicles. *EMS Management Journal, Vol. 1, No. 1*, pages 20–39, 2004.

[15] Jotshi A. and Batta R. Search for an immobile entity on a network. *Submitted to Networks*, 2006.

[16] Yi P., George S. K., Paul J.A., and Lin L. Hospital capacity planning for emergency management in disaster mitigation. *To appear in Socio-Economic and Planning Sciences*, 2006.

APPENDIX

A Interactions for DR federate

Here RG is Report Generator (that generates the reports), ED is Estimate Director (that fuses these reports and gives an estimate of the entity) and DP is Dispatcher/Router.
Input:

(1) *RGtoDP01 - Hospital locations*
This message is received as initialization information and is used to build a list of all hospitals that are available for dispatch.

(2) *EDtoDP01 - Casualty Observation*
This message carries information about a single casualty. It is used to build a catalog of casualties to be used as dispatch targets in the event that no casualty clusters are known. The road node nearest to each hospital is not contained in the message and must be computed by Dispatcher Router.

(3) *EDtoDP02 - Casualty Pickup*
This message indicates that a casualty has been picked up by an ambulance. The message will be ignored and probably dropped from this list.

(4) *EDtoDP03 - Casualty Delivery*
This message indicates that a casualty has been dropped off at a hospital. The message will be ignored and probably dropped from this list.

(5) *EDtoDP04 - Roadway Damage*
This message contains reported information concerning physical damage to a road link. It may represent debris in the road or actual damage to a bridge or a tunnel. This information will be used to compute the average speed with which a vehicle can traverse the link.

(6) *EDtoDP05 - Hospital Capacity*
This message reports a capacity for treatment at a specific hospital, for each of three casualty severities, 1, 2 and 3.

(7) *EDtoDP06 - Travel Delay*
This message carries a reported delay for a specific road link. The delay will change over time and is due to traffic congestion, vehicle accident, large numbers of pedestrians etc.

(8) *EDtoDP07 - Treatment Delay*
This message reports an anticipated treatment delay at a specific hospital, for each of three casualty severities, 1, 2 and 3.

(9) *EDtoDP08 - Ambulance Idle*
This is a report from an ambulance that indicates that the ambulance has no desti-

nation and is available for dispatch. Also reported are the number of severity 2 and severity 3 casualties onboard the vehicle (if any). Different dispatching schemas are used depending on the numbers and severities of onboard casualties.

(10) *EDtoDP09 - Ambulance Stuck*
This report is similar to EDtoDP08 but it further represents the situation where an ambulance has followed its dispatched route, only to find that it is blocked by road link damage that was unknown to the dispatcher at the time of dispatch. The action to be taken will be a re-dispatch that will route the vehicle around the damaged link.

(11) *EDtoDP10 - Cluster Identification*
This is the report from high-level fusion processes of the location where a cluster of casualties has been detected. This cluster and its component cells will be used as dispatch targets because the probability of finding enough casualties (3) to fill an ambulance is high.

Output:
DPtoRG01 - Ambulance Route
This message contains the route from a source (ambulance present location) to a destination (casualty location or hospital). It consists of a series of road nodes interleaved with road links such that there is no 'untraversable' gap in the road map.

B Tele-Atlas

For more information on Tele-Atlas please refer web site *www.na.teleatlas.com*. The input network to our system is given in the form of a text file containing a sequence of records for each link of the network. Each of the links is represented by a 256-character line containing information about the link. The various records associated with a link are:

(1) D Record: Gives the name of the link (road stretch), its two end nodes (intersections of the streets), its length and other major identifying attributes.
(2) A Record: Gives the alternate name of the street (if any).
(3) E Record: Gives the Zip Code and the jurisdiction of the area in which the street is.
(4) S Record: Gives the shape feature of the street, i.e., whether it is a straight road a curvy road.
(5) L Record: Tells us if the link is a bridge or a tunnel and gives it a different ID.

C Attribute Table

Attribute table showing various links and their corresponding attributes is shown in Table C.1.

Table C.1
Attribute table corresponding to various links in Tele-Atlas

Link ID	Name	Length	From ID	To ID	Shape	Class	Bridge	Severity	Delay
26310759	LASSEN ST	14.392	26310762	1352901	0	4	0	0	30
1736047	MARILLA ST	17.254	1736057	1736055	0	4	1	0	70
1409608	PAXTON ST	33.252	1409602	2153734	0	3	1	3	12

D System Level Architectural Strategy

The pseudo-code for the dispatching strategy is as follows.

```
Dispatching()
   For (Each Ambulance)
       if (Ambulance Idle)
           switch (onboard casualty count)
            if (cas == 3)      // ambulance is full
               Dispatch-to-Most-Attractive-Hospital;
               generate route message;
               update state-variables;
               break;        // go to next ambulance
            if (cas == 2)    // room for 1 more
            if (cas == 1)   // room for 2 more
                Step size = 800 (metres)
                if (On-Board ==2)
                   Multiplier = 4
                if (On-Board ==1)
                   Multiplier =  6 or 8 (Depending on severity)
                Dispatch-to-Neighborhood;
               if (dispatch successful)
                   generate route message;
                   update state-variables;
                   break;    // go to next ambulance
                   Dispatch-to-Most-Attractive-Hospital;
                   generate route message;
                   update state-variables;
                   break      // go to next ambulance
            if (cas == 0)      // no casualties onboard
               if (clusters exist)
                  For (Each cluster)
                      calculate attractiveness;
               pick most attractive cluster;
               Dispatch-to-Cluster-Boundary-Cell;
               if (dispatch successful)
                  generate route message;
                  update state-variables;
                  break;      // go to next ambulance
               else           // boundary cells unusable
                  mark cluster "unusable"
                  break;      // repeat this ambulance
             else             // no clusters exist
               Multiplier = 40;
               Dispatch-to-Neighborhood;
           // end of switch
       else
           // this ambulance is moving;
```

Fig. D.1. Pseudocode:Overall Dispatching Strategy

```
Dispatch-to-Neighborhood()
  for (i = 1 to Multiplier)
    neighborhood = i * Step size
    For (Each casualty in neighborhood)
      calculate attractiveness;
    pick casualty with max attractiveness;
    if (max > some minimum attractiveness)
       calculate fastest route;
       report dispatch successful;
    else
       report dispatch unsuccessful;
```

Fig. D.2. Pseudocode:Dispatch to Nearest Casualty

```
Dispatch-to-Cluster-Boundary-Cell()
   find closest cluster;
   For (Each boundary cell in this cluster)
        calculate attractiveness;
   pick max attractiveness;
   if (max > some minimum attractiveness)
      calculate fastest route;
      report dispatch successful;
   else
      report dispatch unsuccessful;
```

Fig. D.3. Pseudocode: Dispatch to Cluster Boundary

```
Dispatch-to-Most-Attractive-Hospital()
   For (Each hospital)
       calculate distance from ambulance loc;
       calculate attractiveness;
   pick most attractive hospital;
   calculate fastest route;
   report dispatch successful;
```

Fig. D.4. Dispatch to Hospital