Routing Service With Real World Severe Weather

YiRu Li[1] Sarah George[1] Craig Apfelbeck[1] Abdeltawab M. Hendawi[1,2]
David Hazel[1] Ankur Teredesai[1] Mohamed Ali[1,3]

[1]*Institute of Technology, University of Washington, Tacoma, WA, USA*
[1]{yiruli, seg6, craigda, hendawi, dhazel, ankurt, mhali}@uw.edu
[2]*Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, USA*
[2]hendawi@cs.umn.edu
[3]*Microsoft Corporation, Redmond, WA, USA*
[3]mali@microsoft.com

ABSTRACT

Traditional routing services aim to save driving time by recommending the shortest path, in terms of distance or time, to travel from a start location to a given destination. However, these methods are relatively static and to a certain extent rely on traffic patterns under relatively normal conditions to calculate and recommend an appropriate route. As such, they do not necessarily translate effectively during severe weather events such as tornadoes. In these scenarios, the guiding principal is not, optimize for travel time, but rather, optimize for survivability of the event, i.e., can we recommend an evacuation route to those users inside the hazardous areas. In this demo, we present a framework for routing services for evacuating and avoiding real world severe weather threats that is able to: (1) Identify the users inside the dangerous region of a severe weather event (2) Recommend an evacuation route to guide the users out to a safe destination or shelter (3) Assure the recommended route to be one of the shortest paths after excluding the risky area (4) Maintain the flow of traffic by normalizing the evacuation on the possible safe routes. During the demo, attendees will be able to use the system interactively through its graphical user interface within a number of different scenarios. They will be able to locate the severe weather events on real time basis in any area in USA and examine detailed information about each event, to issue an evacuation query from an existing dangerous area by identifying a destination location and receiving the routing direction on their mobile devices, to issue an avoidance routing query to ask for a shortest path that avoids the dangerous region, to have an inside look into the internal system components and finally, to evaluate the overall system performance.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Spatial databases and GIS

Keywords

Location Based Services, Evacuation, Avoidance Shortest Path

SIGSPATIAL'14, November 04 - 07 2014, Dallas/Fort Worth, TX, USA

http://dx.doi.org/10.1145/2666310.2666375 ...$15.00.

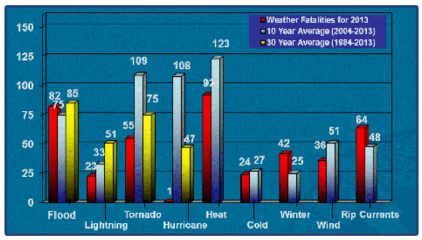

Figure 1: Fatalities Stats by Weather Related Hazards [11]

1. INTRODUCTION

Location-based services aim at saving users time and money. This is offered through a number of widely used spatial queries such as nearest neighbor query, e.g., "find the nearest gas station", and range query, e.g., "recommend an asian restaurant within 2 miles of my location". Routing services, e.g., "find the shortest path from a start location to a destination" is one of the most heavily used location related services, i.e., millions of users visit the routing and mapping websites every day [2]. A special type of routing service during hazardous circumstances focuses on protecting users around the hazards by supporting evacuation and avoidance routing query processing. Evacuation and avoidance routing provides emergency routing service where users' first concern is to save their lives by evacuating from dangerous areas or avoid it in advance. Dangerous areas can be natural, e.g., severe weather, or man-made, e.g., crime venues, events. Here, we focus on weather-related hazards. According to the latest statistics from the national weather service, Figure 1, tornados, floods, hurricanes, and thunderstorms are considered to be the most destructive disasters which result from severe weather events [11, 12]. In this demo paper, we present a framework, named *iTornado*, that supports routing queries while considering existing and predicted severe weather events. On one side, users in a dangerous region, i.e., an area affected by a severe weather, can send an *evacuation routing* query to ask for the shortest path to evacuate to a safe destination of their choosing, outside of that risky region. At the other side, users outside a dangerous region can send an *avoidance routing* query to obtain a shortest path that avoids crossing this area. For brevity, through the rest of the paper we might use *tornado* as a surrogate for any dangerous weather event, and *tornado region* to refer to the dangerous area created by this event.

The main idea of the proposed system is to access the warnings information from the National Weather Service, and to identify those areas that are expected to be affected by a severe weather event such as a tornado. The system then recommends *evacuation routes* to the moving objects, i.e., vehicles, inside hazardous areas,

and *avoidance routes* to the objects around these areas. This problem is challenging due to two facts. (1) Due to the speed at which severe weather patterns develop and move, as well as the potential for loss of life if routing is incorrect or delayed, these special types of routes have to be computed with very low latency while maintaining very high precision. (2) Applying pure shortest path algorithms with obstacle avoidance [10] might cause a traffic jam to the surrounding roads which in turn negatively affects the traffic flow in these roads. The proposed *iTornado* system is equipped with a set of data structures and algorithms to tackle these challenges as follows. Through constant communication with the National Weather Service web service API, the *iTornado* system receives frequent updates to weather events. Then, a grid data structure is employed to represent the affected area on the underlying space. We use fine resolution cell size to assure more accurate coverage of the tornado's exact region. This helps to trim down the affected area, in turn reducing the number of objects that are expected to send evacuation queries. To compensate for any unexpected change in the tornado shape, speed, or direction, we add a marginal space around the original tornado region. When combined with the tornado region, we refer to this larger zone as the *risky region*. For the objects inside the *risky region*, we predict their next destination within a specific time period t. Where t equals to the time required from the object to travel from its current location to the nearest boundary of the *tornado region*. If the object's destination is expected to touch the tornado region, then we ask that object to release its destination to form an evacuation routing query. If it is not the case, the object is not alerted. Instead, we wait until a serious change happens to its ongoing movements. The employed prediction model is based on the assumption that objects tend to follow efficient routes in their travel to destinations [5, 7]. To avoid causing a traffic jam, when the system recommends evacuation route, we distribute the load on the K convenient roads rather than recommending the same road for many objects. We weigh the capacity of each road by its speed limit. For example, a road with a 70 miles per hour speed limit can be suggested to seven contiguous objects while the one with 40 miles per hour can be suggested to four objects.

2. PROBLEM SETTINGS

In this section, we define the research problem addressed in this paper. Then we set the framing assumptions.

2.1 Problem Definition

The research problem studied in this paper can be formalized as follows. Given, (1) a set of moving objects $\mathcal{O}$, (2) a road network graph $G(N, E, W)$, where N is the set of nodes, E is the set of edges, and W is the edge weights, and (3) a set of prediction results of severe weather events (list of affected areas, start time, end time for each area, shape as polygon,...). For example, a warning for thunderstorm prediction can be like this *"severe thunderstorm near Eufaula, by 5:35PM lat 31.99 lon 85.05, etc"* [11].

We aim to answer evacuation queries by recommending evacuation shortest paths to the objects inside the areas affected by severe weather events to their destinations, as well as avoidance routing queries to help the objects around the risky areas to travel to their destination without crossing these areas. Our objective is to

(1) reduce the response time (2) reduce the the number of objects receiving alerts. and (3) reduce the possibility of causing a traffic jam in the surrounding area.

2.2 Assumption

We assume that moving objects will be connected to the *iTornado* evacuation routing server and keep updating their location periodically. In addition, we assume that moving objects mostly travel through efficient routing behavior. This means each step an object takes ahead makes it closer to its destination and farther from its start location. This means, we do not assume that we have access to objects historical trajectories. Prediction of objects' next destinations is based solely on the knowledge obtained from their current trips, i.e., sequence of passed locations [4, 6, 5, 7, 8]. For the severe weather prediction and alarming, we assume that the prediction results will be obtained from the National Weather Service by accessing the National Digital Forecast Database (NDFD) in addition to the tornadoes warnings stream. We need to emphasize here that our objective is to leverage the severe weather forecast service to provide real time evacuation and avoidance routing service. This service aims at helping those moving objects inside the hazardous region to evacuate safely to their destinations and those outside that region to travel without passing through it.

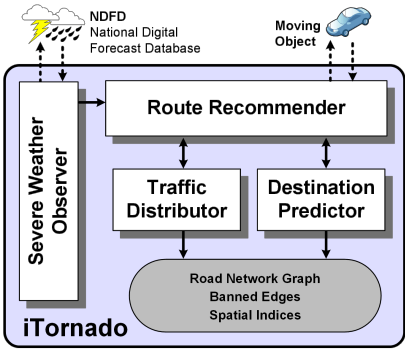

Figure 2: iTornado System Architecture

3. SYSTEM OVERVIEW

In this section we give an overview for the proposed system. We briefly describe the system's main data structures and modules.

Figure 2 gives the architecture of the proposed *iTornado* framework, which consists of four main modules, namely, the *severe weather observer* module, the *route recommender* module, the *destination predictor* module, and the *traffic distributor* module. A concise description of the leveraged data structures as well as each of these modules will be provided in the following sections.

3.1 Data Structures

In the *iTornado* framework, we have three basic data structures to maintain: (1) *Road Network Graph*, which contains a set of nodes V and edges E, where the edge weights represent the travel time. (2) *Banned Edges*, which carries the list of edges that intersect with the risky region, i.e., warning polygon. We use it to serve the computation of the shortest path that avoids the risky region. (3) *Grid index*, inside the destination predictor module, we maintain a grid data structure to obtain a cell based prediction for the objects next movements. The same data structure is leveraged to represent the risky region. Here, we need to notice that, the smaller the cell size, the better representation of the warning polygon and in turn, the less evacuation space to consider. (4) *Paths buffer*, for each cell in the risk region, we carry a list of the K alternative shortest paths to evacuate from the current location to the nearest safe cell outside that region.

3.2 Severe Weather Observer

To get timely forecasts and warnings updates, the *severe weather observer* module periodically queries the NDFD for the area of concern [11] as well as consumes the tornadoes warnings feeds.

Through sending SOAP (simple object access protocol) web service requests, we get access to official national weather forecasts computed by forecasters at local weather offices and national centers as well. To divide the space down into small areas of forecast, they have a preset grid spatial resolutions ranging from 1.25 to 10 kilometer per tile. Each request can obtain rich data about the existing severe weather warning which includes, location of the warning, the shape as polygon, speed, direction, the list of areas expected to be affected, and the start and end times.

3.3 Destination Predictor

This module aims at anticipating the future movements of a moving object in the area around the severe weather. The purpose of this module is to evaluate which objects to exclude from evacuation alerts, those that are expected to move outside the risky region. Also, for those objects that are near the severe weather range but not exactly in the risk region, we need to know if they are expected to pass by that region or not. If this is not the case, there is no need to waste the system resources to compute alternate paths for them. Otherwise, we ask for their destinations and run an avoidance shortest path query to get the route that avoids the severe weather region. The employed prediction model divides the space into a number of cells and takes a sequence of passed cells by an object in its current trip. Then, it computes the probability of a certain cell being a possible destination based on the number of cells accessible through an efficient route [5, 7]. Here, efficient route means each object's step should bring it closer to its destination.

3.4 Traffic Distributor

This module assures the fair distribution of the recommended evacuation routes over the convenient roads in the underlying road network. Basically, the higher the speed of a road the larger the number of times it is recommended to users. In other words, much weight is given to highways, as they can serve more moving objects than surface roads. Once a route exceeds its maximum number of times to be recommended to users, we switch to the next best candidate. This is done by leveraging the *paths buffer* to loop on the convenient evacuation routes from the object's current cell to the nearest safe cell.

3.5 Route Recommender

Based on the warning information we get from the *severe weather observer*, we apply the following steps. Firstly, we get the list of cells that overlap with the warning polygon. These cells represent the severe weather event, named *tornado region*. Then, we add to it a marginal space buffer to form the *risky region*. by doing this, we take preventive action to keep objects from straying into the *tornado region*. The *risky region* is used to guarantee that the suggested avoidance route will be far enough from the heart of hazards. Next, we identify the objects that are actually inside the *tornado region*, and send them an evacuation alert. This is done by visiting the grid index to get the list of objects inside each cell overlaps that region. Based on the destination an objects wants to evacuate to, we find the shortest path to the nearest safe node outside the *risky region*. From there, we find the list of edges that touch, intersect or are contained within the *risky region* and set its weight to infinite. Computing the shortest path [9] given the updated road network graph will result in a shortest path that does not cross the *risky region*. A copy of the original edges before changing their costs will be held in the *banned edges* list. This list will be used to recover the road network graph in this area after the tornado passes and any debris has been cleared.

To reduce the number of objects to evacuate, we (1) represent the tornado region by a set of small rectangles rather than one big rectangle, and (2) we evacuate those objects that are actually inside the *tornado region*, or inside the *risky region* and tend to go toward the *tornado region*. For those objects that appear to miss the *risky region*, we postpone the evacuation alert. To reduce the likelihood of impacting traffic on the suggested routes, the *route recommender* examines the *traffic distributer* and modifies its recommendation accordingly.

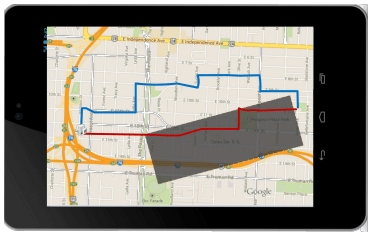

Figure 3: User Mobile Interface

4. DEMO SCENARIOS

This section illustrates the possible scenarios of using the proposed *iTornado* system. The audience will be able to interact with the system through a set of accessible graphical user interfaces. Throughout the demo, we explain how to communicate to the system from the end user's perspective, and from the system inspector perspective. The earlier view depicts how users can send routing queries and receive the results, while the later zooms inside the internal components to show how the queries are processed using the employed data structures and algorithms. For demonstration purposes, we use real data sets for a number of tornados in Kansas, e.g., tornado on 2012/4/14 from 16:33PM to 18:31PM, extracted from the National Digital Forecast Database (NDFD) using the National Weather Service API. To represent the moving objects around the tornado areas, we leverage the Brinkhoff's generator [3] to obtain a large set of synthetic data of moving objects. To visualize what is happening on the computation side, we employ the World Wide Telescope (WWT) as our visualization engine [1]. The following different scenarios will be provided during the demonstration venue.

4.1 Scenario 1: Avoidance Routing

In this scenario, the audience can follow the object (a vehicle) that is moving on the map as a tornado touches down near it. The object is not in the danger zone yet, but its destination currently requires it to cross the danger zone. Here we re-route the object around the tornado danger zone to its destination. Figure 3 demonstrates a mobile user interface that will receive and display the alternative routes to avoid the severe weather risk region (grey rectangular area). The red path represents the original shortest path that crosses the risky area, while the blue path is our recommended path. This path should be one of the Kth shortest paths that come second after the original risky one, in terms of travel time or distance. Both paths are connecting the moving object - the green car - and its destination, the visible white building.

4.2 Scenario 2: Evacuation Routing

In the evacuation routing scenario, the audience will be able to notice the objects that are exactly inside the *tornado region*, red cars inside red bounded cells in Figure 4. For each of these objects, an evacuation route will be suggested by finding the shortest path to the nearest safe node outside the tornado region. Then, we compute the shortest path from that node to the object's destination while avoiding the tornado region. The merge of those two paths will

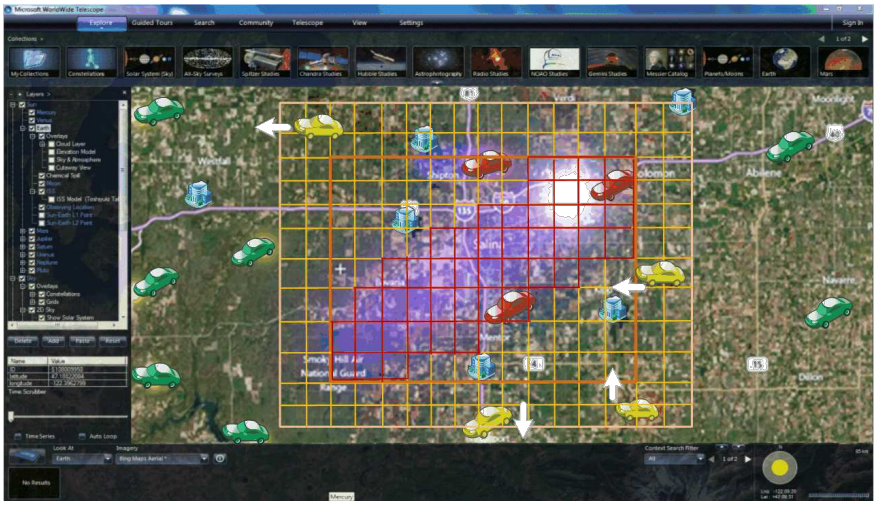

Figure 4: The Server Insight Interface

form the evacuation route.

4.3 Scenario 3: Predictive Evacuation Routing

This scenario involves predicting the possible destinations of the moving objects at the *risky region*, but not exactly inside the *tornado region*. For each of these objects, we predict its next movements based on the previous locations in its current trip, the object current speed, and its direction. This will allow us to predict where the vehicle may go in the next few minutes. When the moving objects in this scenario are visualized, we add an arrow to the car icon which indicates whether the car is more likely to complete its trip toward the tornado region or away from it, Figure 4. Thus, we focus to alert only those objects that tend to go closer to the *tornado region*.

4.4 Scenario 4: Exploration

A user can draw a rectangle on the area of interest on the map to send an exploration range query to our system. The *iTornado* should go and query the NDFD and get the forecasts for the given range. The result will return the existing alerts, if any, for the selected area within the specified time frame in the future. The results include specific pieces of information, for example, the type of the alert, its exact coordinates, start time and end time, direction, speed, and severity. In addition to that, the results can be visualized to highlight the dangerous spots.

4.5 Scenario 5: System Insight

The audience will be able to examine the grid structure and identify those cells that directly intersect with the *tornado region* and those inside the *risky region* around the tornado. As shown in Figure 4, the cell bounded by red represents the *tornado region*, while those with yellow boundaries represent the *risky region*. They will also figure out how the system starts by the objects inside the *tornado region*, red cars. They will be directed to outside both tornado and risky ranges to either a destination of their selection or to the closest safe shelter, blue buildings. Moreover, the users will examine how the prediction model is applied on those objects inside the *risky region*, but outside the *tornado region* to have a better prioritization for whom should be evacuated first.

4.6 Scenario 6: Stress Test

In this scenario, the audience will be able to test the efficiency and scalability of the proposed *iTornado* framework. This will be achieved by executing a large number of different avoidance routing and evacuation requests, rather than a single request at a time as in the above scenarios. In addition, this scenario will illustrate the performance of the system with a large number of severe weather areas. Those areas are represented by a set of synthetically generated rectangular regions distributed over the underlying map. Furthermore, the audience will have the opportunity to test the scalability of the system with large numbers of moving objects prepared using the Brinkhoff generator. Finally, the audience will be able to inspect the produced graphs and charts resulted from the performance tests.

5. REFERENCES

[1] M. Ali, B. Chandramouli, J. Fay, C. Wong, S. Drucker, and B. S. Raman. Online Visualization of Geospatial Stream Data using the WorldWide Telescope. In *VLDB*, Washington, USA, Aug. 2011.

[2] AllWebsiteStats. Statisitcs For Websites Usage. `http://allwebsitestats.com/`, June 2014.

[3] T. Brinkhoff. A Framework for Generating Network-Based Moving Objects. *GeoInformatica*, 6(2):153–180, 2002.

[4] A. M. Hendawi, J. Bao, and M. F. Mokbel. iRoad: A Framework For Scalable Predictive Query Processing On Road Networks. In *VLDB*, Riva Del Garda, Italy, Aug. 2013.

[5] A. M. Hendawi and M. F. Mokbel. Panda: A Predictive Spatio-Temporal Query Processor. In *ACM SIGSPATIAL GIS*, California, USA, Nov. 2012.

[6] A. M. Hendawi and M. F. Mokbel. Predictive Spatio-Temporal Queries: A Comprehensive Survey and Future Directions. In *MobiGIS*, California, USA, Nov. 2012.

[7] J. Krumm. Real Time Destination Prediction Based on Efficient Routes. In *SAE*, Michigan, USA, Apr. 2006.

[8] J. Krumm, R. Gruen, and D. Delling. From Destiantion Prediction To Route Prediction. *Technical Report. Microsoft Research*, 2011.

[9] H. P.E., N. N.J., and R. B. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[10] M. Rice and V. J. Tsotras. Graph Indexing of Road Networks for Shortest Path Queries with Label Restrictions. *PVLDB*, 4(2):69–80, 2010.

[11] N. W. Service. National Digital Forecast Database. `http://www.nws.noaa.gov`, June 2014.

[12] Wikipedia. Severe Weather. `http://en.wikipedia.org/wiki/Severe_weather`, June 2014.