Dynamic Emergency Ambulance Fleet Allocation
Master Thesis

Niklas Meinzer

University of Freiburg
Faculty of Engineering

May 14, 2014
Motivation

Motivation

Emergency Medical Services (EMS)

- Part of healthcare systems
- Ambulances respond to emergency calls
- Treat patients and transport them to a hospital

- What strategies can be used to manage a fleet of ambulances?
- How can we measure their performance?
Problem

Definition

Given a graph $G = (E, V)$, a set of hospitals $H \in V$, a set of ambulance bases $B \in V$, a fleet of ambulances and a stream of requests, maximise the number of saved patients.

Secondary goals:

- Minimise the response time, i.e., the time between an emergency call and the arrival of an ambulance at the patient.
- Try to keep cost of operations low.

$\Rightarrow$ To develop strategies we need a model.
We developed the following model for the EMS setting:

**Graph**

- A road network graph serves as base for the model.
- Edges/Roads are undirected
- All roads are of the same type
We developed the following model for the EMS setting:

<table>
<thead>
<tr>
<th>Graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>- A road network graph serves as base for the model.</td>
</tr>
<tr>
<td>- Edges/Roads are undirected</td>
</tr>
<tr>
<td>- All roads are of the same type</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Hospitals</td>
</tr>
<tr>
<td>- Ambulance Bases</td>
</tr>
<tr>
<td>- Ambulances</td>
</tr>
</tbody>
</table>
Requests

Requests represent emergency calls by a patient anywhere on the graph and consist of:

- The time of the call
- The location of the patient
- The time-to-live (TTL) of the patient.

Both time of call and location of patient are chosen at random.

Agents

Agents represent EMS operators and are able to give orders to ambulances.
Simulation

- Event driven simulation is used to test different strategies.
- Events are kept in a priority queue.
Agents are informed about the following events:

- EMS Request
- Ambulance at Patient
- Ambulance Free

When making decisions it can:

- See ambulances position and state
- See current and past requests
- Run (route planning) algorithms
- Give orders to ambulances

How can the strategies influence the outcome?
Strategies - Greedy

First let’s try the easy way: Only move ambulances around if absolutely necessary

<table>
<thead>
<tr>
<th>Actions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>New Requests</strong></td>
<td>⇒ Send closest ambulance</td>
</tr>
<tr>
<td><strong>Ambulance at patient</strong></td>
<td>⇒ Send ambulance to closest hospital</td>
</tr>
<tr>
<td><strong>Ambulance Free</strong></td>
<td>⇒ Send ambulance back to base</td>
</tr>
</tbody>
</table>
First let’s try the easy way: Only move ambulances around if absolutely necessary

**Actions**

| New Requests | ⇒ Send closest ambulance |
| Ambulance at patient | ⇒ Send ambulance to closest hospital |
| Ambulance Free | ⇒ Send ambulance back to base |

**Characteristics**

- Computationally cheap
- Low overall travel distances
- Very dependant on prior ambulance placement
- Used in practice
Now: Attempt to maintain an optimal distribution of ambulances throughout the graph.

K-Medoids

Similar to K-Means:
Finds a clustering around k elements of the input set, that minimizes the average distance of a node to its medoid.
Now: Attempt to maintain an optimal distribution of ambulances throughout the graph.

**K-Medoids**

Similar to K-Means:
Finds a clustering around $k$ elements of the input set, that minimizes the average distance of a node to its medoid.

Idea: When we have $k$ free ambulances, run $k$-Medoid and position the ambulances at all the medoids.
Strategies - K-Medoid

Niklas Meinzer
Dynamic Emergency Ambulance Fleet Allocation
Strategies - K-Medoid
**Strategies - K-Medoid**

<table>
<thead>
<tr>
<th>Actions</th>
<th>Action Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Start</strong></td>
<td>Redistribute all ambulances according to K-Medoid</td>
</tr>
<tr>
<td><strong>New Requests</strong></td>
<td>Send closest ambulance to request and redistribute free ambulances</td>
</tr>
<tr>
<td><strong>Ambulance at patient</strong></td>
<td>Send ambulance to closest hospital</td>
</tr>
<tr>
<td><strong>Ambulance Free</strong></td>
<td>Redistribute free ambulances</td>
</tr>
</tbody>
</table>

Characteristics:
- Computationally more complex, needs precomputation
- Probably very high overall distances, but we hope for good response times
- No dependency on prior ambulance placement
- Not very practical

Niklas Meinzer
Dynamic Emergency Ambulance Fleet Allocation
Motivation
Problem and Model
Strategies
Results

Strategies - K-Medoid

Actions

Start ⇒ Redistribute all ambulances according to K-Medoid
New Requests ⇒ Send closest ambulance to request and redistribute free ambulances
Ambulance at patient ⇒ Send ambulance to closest hospital
Ambulance Free ⇒ Redistribute free ambulances

Characteristics

- Computationally more complex, needs precomputation
- Probably very high overall distances, but we hope for good response times
- No dependency on prior ambulance placement
- Not very practical
Greedy and K-Medoid both have downsides. Can we find a compromise?

**Voronoï**

Based on Voronoï diagrams (see next slide).

Idle ambulances are only allowed to be positioned at ambulance bases or hospitals.
Motivation

Problem and Model

Strategies

Results

Strategies - Voronoi

Greedy and K-Medoid both have downsides. Can we find a compromise?

Voronoi

Based on Voronoi diagrams (see next slide).
Idle ambulances are only allowed to be positioned at ambulance bases or hospitals.

Idea: Divide the graph into Voronoi-cells around the ambulance bases and hospitals and distribute ambulances according to the cluster size.
Motivation
Problem and Model
Strategies
Results

Strategies - Voronoi

Niklas Meinzer
Dynamic Emergency Ambulance Fleet Allocation
## Strategies - Voronoi

<table>
<thead>
<tr>
<th>Actions</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Start</strong></td>
<td>Redistribute all ambulances according to size of Voronoi cells</td>
</tr>
<tr>
<td><strong>New Requests</strong></td>
<td>Send closest ambulance to request and redistribute free ambulances</td>
</tr>
<tr>
<td><strong>Ambulance at patient</strong></td>
<td>Send ambulance to closest hospital</td>
</tr>
<tr>
<td><strong>Ambulance Free</strong></td>
<td>Redistribute free ambulances</td>
</tr>
</tbody>
</table>
Strategies - Voronoi

Actions

- **Start** ⇒ Redistribute all ambulances according to size of Voronoi cells
- **New Requests** ⇒ Send closest ambulance to request and redistribute free ambulances
- **Ambulance at patient** ⇒ Send ambulance to closest hospital
- **Ambulance Free** ⇒ Redistribute free ambulances

Characteristics

- Less complex precomputation and redistribution logic
- Travel distances lower, because not all ambulances move every time
- Somewhat dependant on object distribution
- More practical than K-Medoid
Dynamic Reassignment

Until now: Ambulances are assigned to requests in a *send and forget* manner: Once an ambulance is assigned to a request this assignment will not change.

- Is that bad?
- How can we fix it?
Dynamic Reassignment

Niklas Meinzer
Dynamic Emergency Ambulance Fleet Allocation
Dynamic Reassignment
Dynamic Reassignment

Niklas Meinzer
Dynamic Emergency Ambulance Fleet Allocation
Dynamic Reassignment
Dynamic Reassignment

A

B

C

D
To solve this we use the following reassignment strategy:

Once a new request comes in:

- Take the set of all free ambulances
- And all open requests
- Compute travel costs for each ambulance to all patients
- Compute optimal assignment using the Hungarian Algorithm

⇒ (Re-)Assign ambulances accordingly
Evaluation
Evaluation

- Average number of patients saved versus expected time between requests (seconds)
- Average response time (seconds)

Graphs show performance comparison between Greedy, K-medoid, and Voronoi strategies.
Evaluation

![Graph showing the total distance driven (kilometers) against expected time between requests (seconds) for different strategies: Greedy, K-medoid, and Voronoi. The graph illustrates that the K-medoid strategy results in the lowest total distance driven compared to the other strategies.](image-url)
Evaluation
Evaluation

![Graph showing evaluation results for different strategies. The x-axis represents expected time between requests (seconds), and the y-axis represents the number of patients saved or response time (seconds). The graph compares the performance of Greedy, K-medoid, and Voronoi strategies.]
Evaluation

The graph shows the total distance driven (in kilometers) over time, with the y-axis representing distance and the x-axis representing the expected time between requests (in seconds). Three strategies are compared:

- **Greedy**: represented in red.
- **K-medoid**: represented in blue.
- **Voronoi**: represented in green.

From the graph, it is evident that the K-medoid strategy maintains a lower total distance driven compared to the Greedy and Voronoi strategies, particularly as the expected time between requests increases.
Evaluation

With Reassignment:

![Graphs showing comparison between different strategies]

- Greedy
- K-medoid
- Voronoi
- RA Greedy
- RA K-medoid
- RA Voronoi
K-Medoids shows advantages over a regular Greedy approach, but is uneconomic

Voronoi as a more realistic compromise

Advantages over Greedy especially in unfavourable environments

Reassignment improves performance of all approaches