Dynamic Emergency Ambulance Fleet Allocation

Master Thesis

Niklas Meinzer

University of Freiburg
Faculty of Engineering

May 14, 2014

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:

- Minimize 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.

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

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

Objects

- Hospitals
- Ambulance Bases
- Ambulances

Model

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.

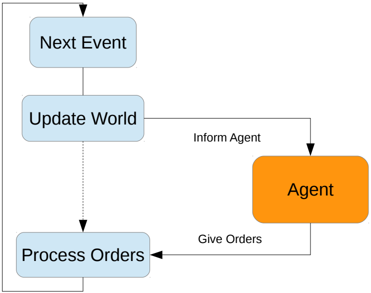
Both time of call and location of patient are chosen at random.

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



Strategies - Overview

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

Actions

New Requests	$\Rightarrow$	Send closest ambulance
Ambulance at patient	$\Rightarrow$	Send ambulance to closest hospital
Ambulance Free	$\Rightarrow$	Send ambulance back to base

Strategies - Greedy

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

Actions

New Requests	$\Rightarrow$	Send closest ambulance
Ambulance at patient	$\Rightarrow$	Send ambulance to closest hospital
Ambulance Free	$\Rightarrow$	Send ambulance back to base

Characteristics

- Computationally cheap
- Low overall travel distances
- Very dependant on prior ambulance placement
- Used in practice

Strategies - K-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.

Strategies - K-Medoid

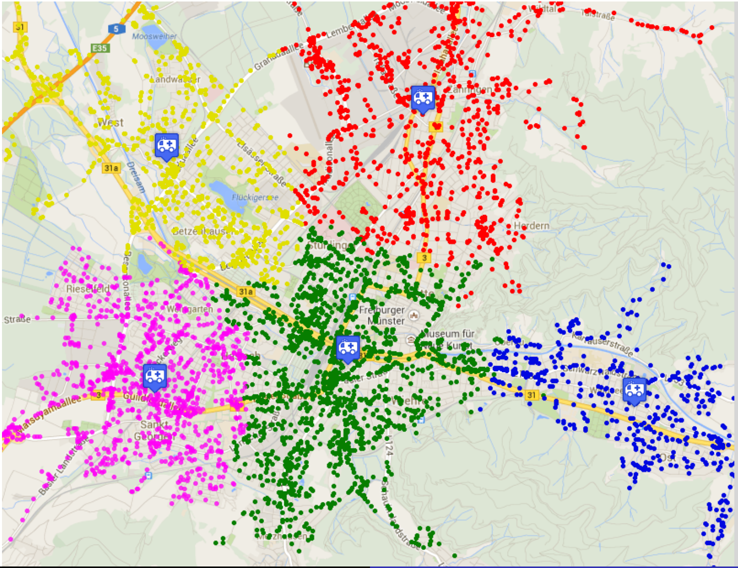
Now: Attempt to maintain an optimal distribution of ambulances throughout the graph.

K-Medoids

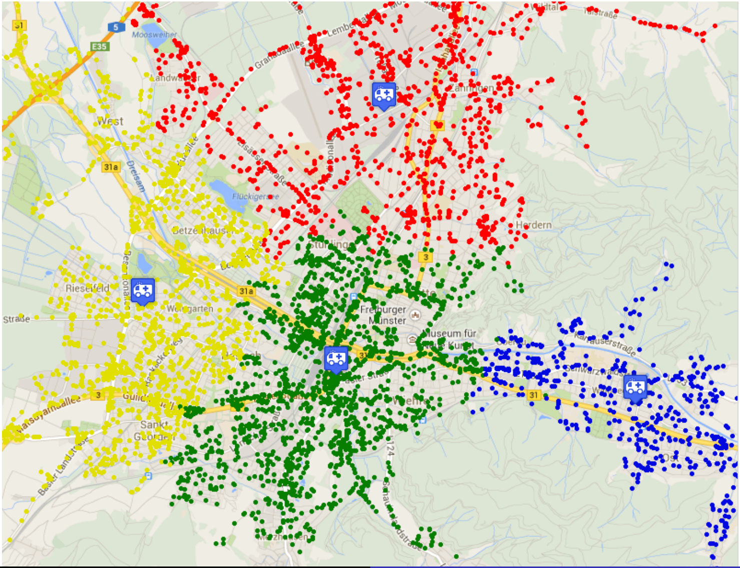
Similar to K-Means:
Finds a clustering around k elements of the input set, that minimizes the average distance of a node to its medoid.

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

Strategies - K-Medoid



Strategies - K-Medoid



Strategies - K-Medoid

Actions

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

Strategies - K-Medoid

Actions

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

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.

Strategies - Voronoi

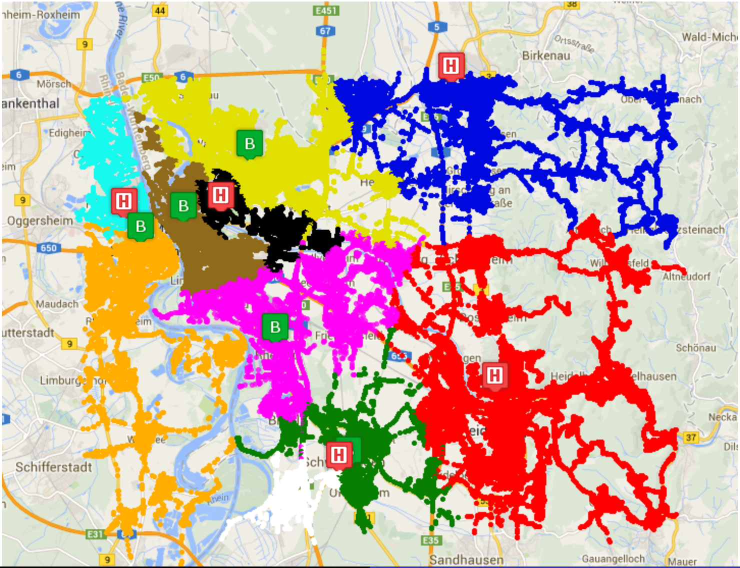
Greedy and K-Medoid both have downsides. Can we find a compromise?

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.

Strategies - Voronoi



Strategies - Voronoi

Actions

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

Strategies - Voronoi

Actions

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

Characteristics

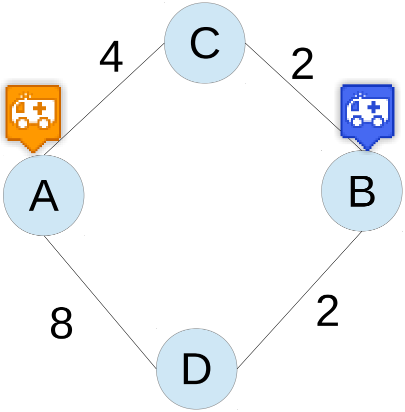
- Less complex precomputation and redistribution logic
- Travel distances lower, because not all ambulances move every time
- Somewhat dependant on object distribution
- More practical than K-Medoid

Dynamic Reassignment

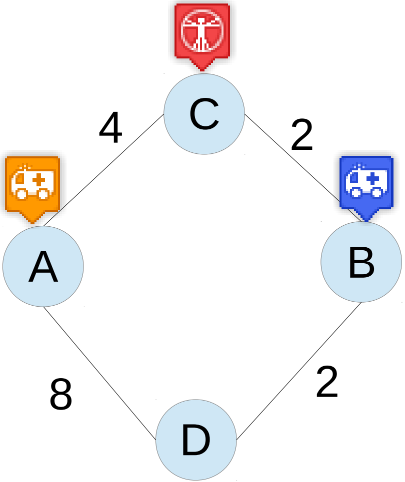
Until now: Ambulances are assigned to requests in a *send and forget* manner: Once an ambulance is assigned to a request this assignment will not change.

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

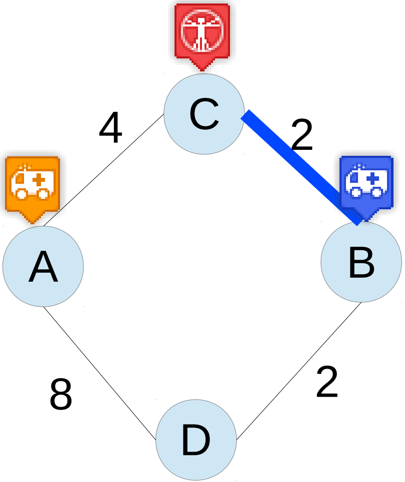
Dynamic Reassignment



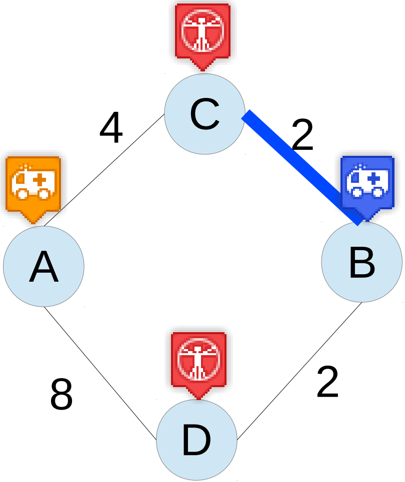
Dynamic Reassignment



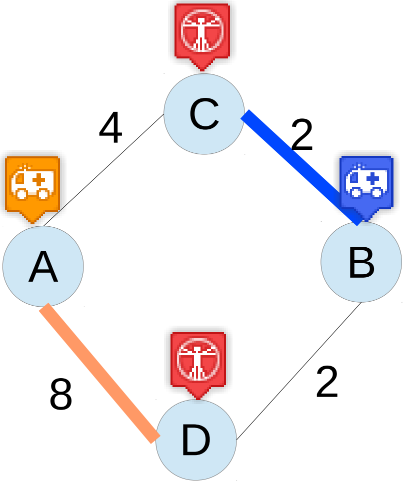
Dynamic Reassignment



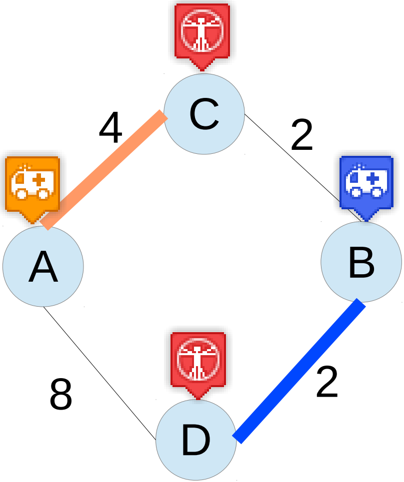
Dynamic Reassignment



Dynamic Reassignment



Dynamic Reassignment



Dynamic Reassignment

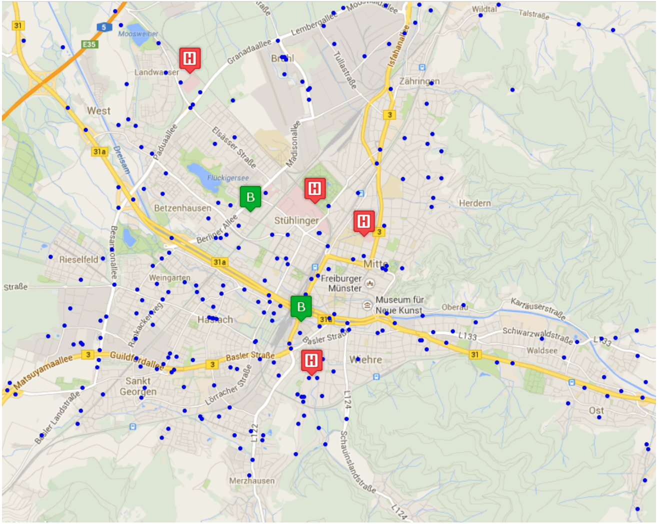
To solve this we use the following reassignment strategy:

Once a new request comes in:

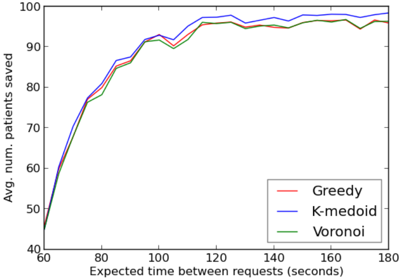
- Take the set of all free ambulances
- And all open requests
- Compute travel costs for each ambulance to all patients
- Compute optimal assignment using the Hungarian Algorithm

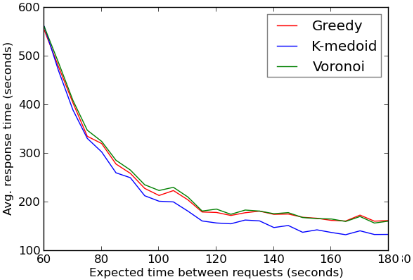
$\Rightarrow$ (Re-)Assign ambulances accordingly

Evaluation

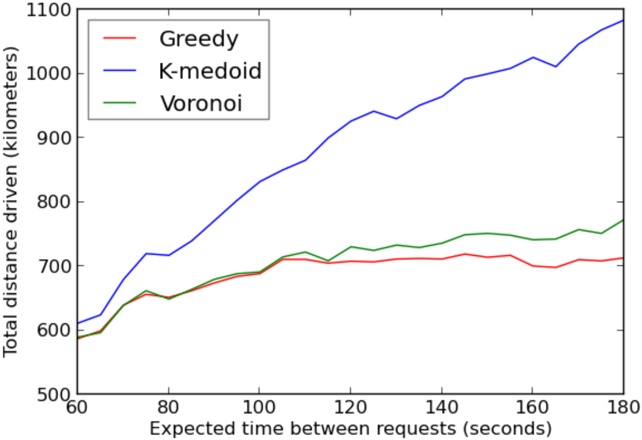

West
Landwasser
Brühl
Zähringen
Betzenhausen
Stühlinger
Herdern
Rieselfeld
Weingarten
Mitte
Freiburger Münster
Museum für Neue Kunst
Haslach
Oberau
Sankt Georgen
Wiehre
Waldsee
Ost
Merzhausen
Flückigersee
Moosweiher
Granadaallee
Lembergallee
Paduaallee
Elsässer Straße
Madisonallee
Tullastraße
Isfahanallee
Berliner Allee
Besançonallee
Dreisam
Basler Straße
Guildfordallee
Matsuyamaallee
Lörracher Straße
Schauinslandstraße
Schwarzwaldstraße
Kartäuserstraße
Wildtal
Talstraße
Basler Landstraße
Straße
B
H

Evaluation

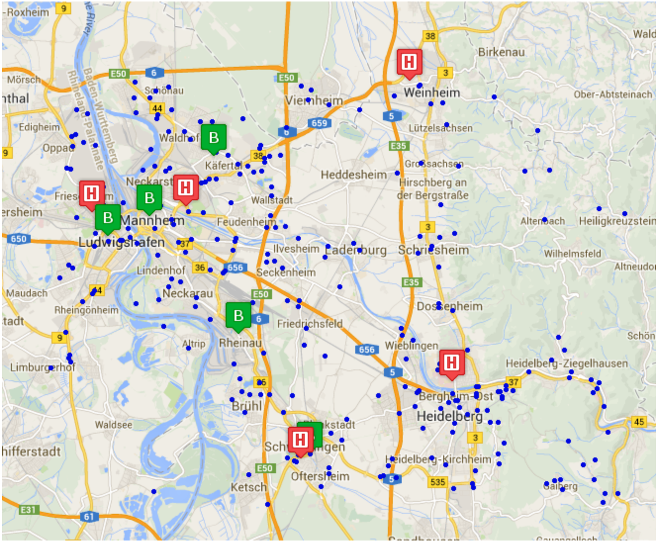

Avg. num. patients saved
100
90
80
70
60
50
40
60
80
100
120
140
160
180
Expected time between requests (seconds)
Greedy
K-medoid
Voronoi


Avg. response time (seconds)
600
500
400
300
200
100
60
80
100
120
140
160
180
Expected time between requests (seconds)
Greedy
K-medoid
Voronoi

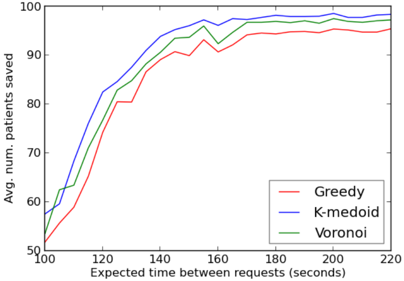
Evaluation

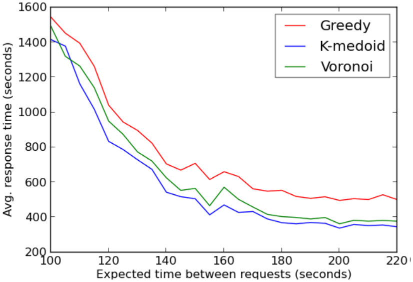

1100
1000
900
800
700
600
500
Total distance driven (kilometers)
60
80
100
120
140
160
180
Expected time between requests (seconds)
Greedy
K-medoid
Voronoi

Evaluation

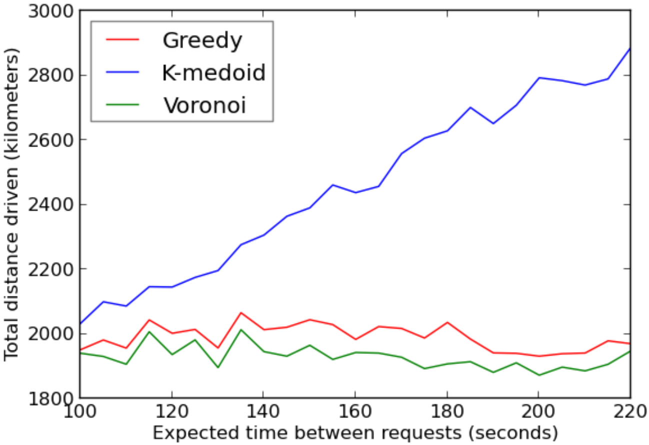

Weinheim
Birkenau
Viernheim
Lützelsachsen
Großsachsen
Heddesheim
Hirschberg an der Bergstraße
Käfertal
Wallstadt
Mannheim
Ludwigshafen
Feudenheim
Ilvesheim
Ladenburg
Schriesheim
Seckenheim
Lindenhof
Neckarau
Friedrichsfeld
Dossenheim
Wieblingen
Rheinau
Altrip
Brühl
Heidelberg
Bergheim-Ost
Heidelberg-Ziegelhausen
Heidelberg-Kirchheim
Oftersheim
Ketsch
Waldsee
Limburgerhof
Rheingönheim
Maudach
Edigheim
Oppau
Mörsch
Schönau
Waldhof
Altenbach
Heiligkreuzstein
Wilhelmsfeld
Ober-Abtsteinach
Gaiberg

Evaluation


100
90
80
70
60
50
Avg. num. patients saved
100
120
140
160
180
200
220
Expected time between requests (seconds)
Greedy
K-medoid
Voronoi

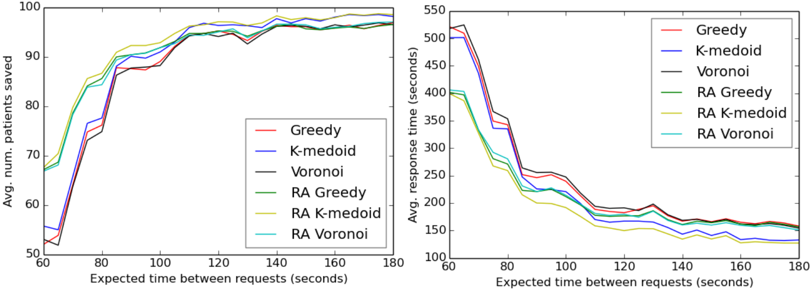

1600
1400
1200
1000
800
600
400
200
Avg. response time (seconds)
100
120
140
160
180
200
220
Expected time between requests (seconds)
Greedy
K-medoid
Voronoi

Evaluation


Greedy
K-medoid
Voronoi
Total distance driven (kilometers)
Expected time between requests (seconds)
3000
2800
2600
2400
2200
2000
1800
100
120
140
160
180
200
220

Evaluation

With Reassignment:



Conclusion

- 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