Routing for Relief Efforts

Minimising Response Time in Natural Disasters

Gefördert im Rahmen der Exzellenzstrategie von Bund und Ländern

Presenter: Marvin Beese

TU Berlin

Supervisor: Dr. Martin Aleksandrov

FU Berlin - Dahlem Center for Machine Learning and Robotics

Date: 25.09.2023

Sturop: Berlin Conference for Student Research 2023, Berlin University Alliance

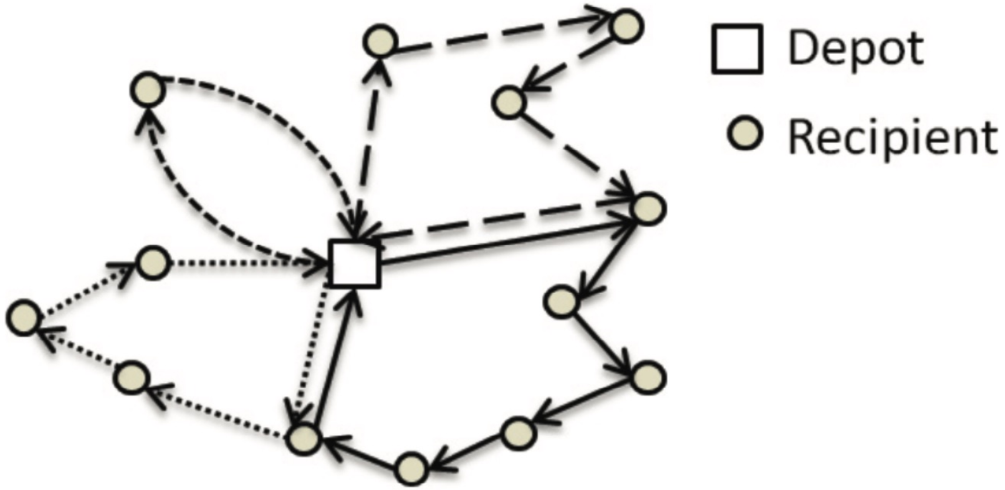
Outline

- Motivation
- Research Question
- Method
- Experiments
- Conclusion

Motivation

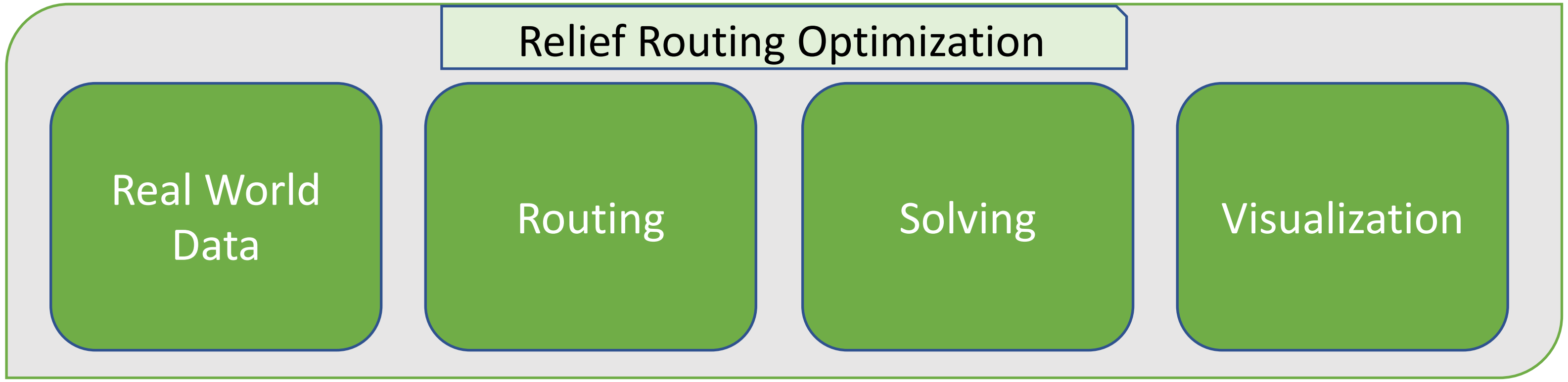
- Rescue of people during natural disasters
- Minimizing Response Time
- Vehicle Routing Problem

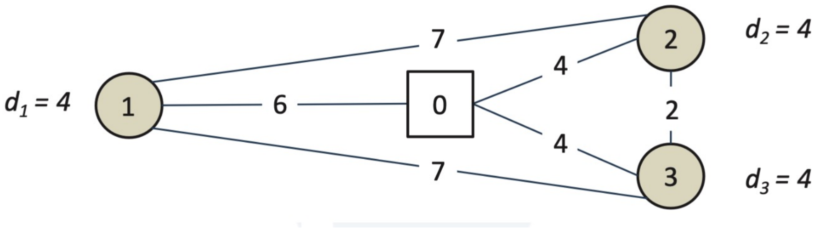
Turkey Earthquake 2023 Destruction [im1]



Last-Mile Delivery Problem Instance Example [1]

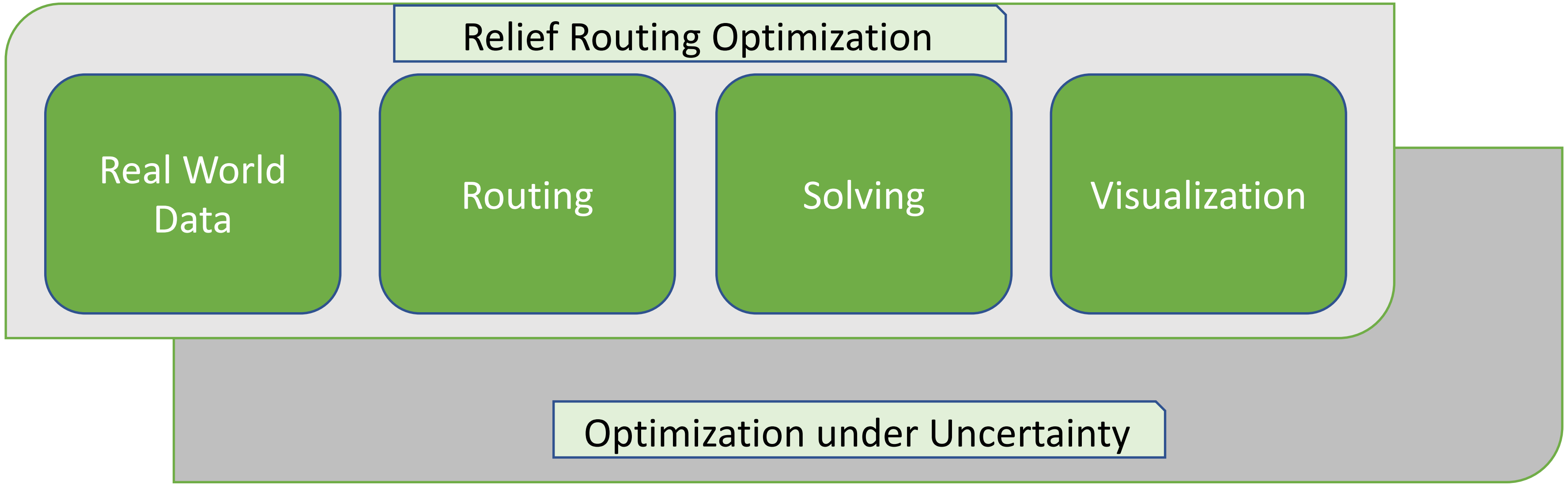
Research Question

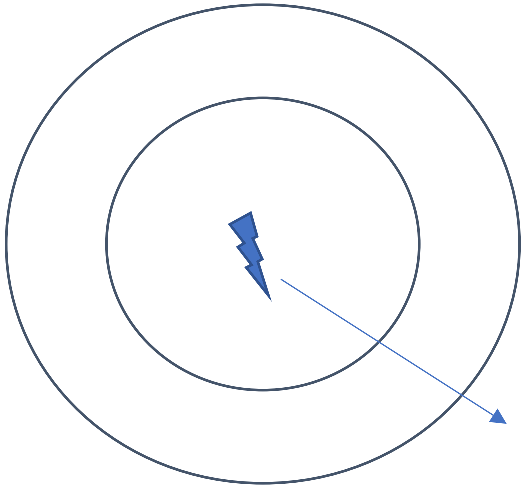




Distribution Network with arc lengths [1]

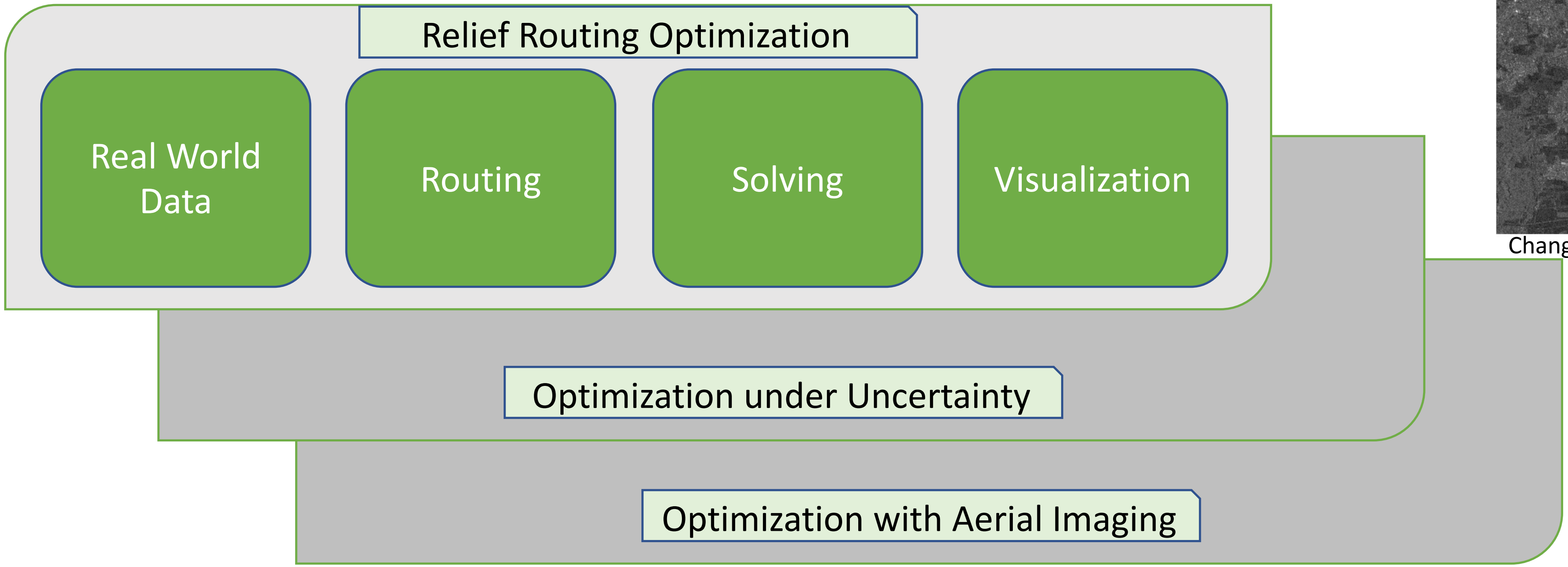
Research Question

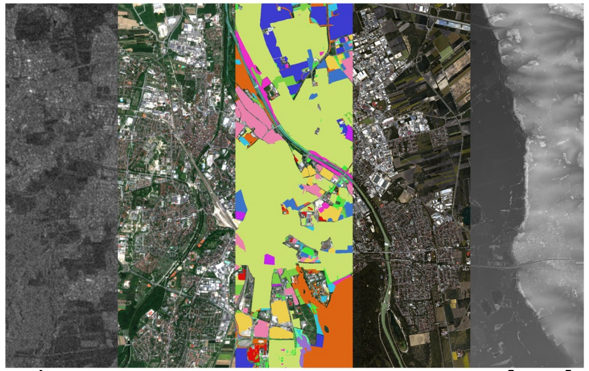




$$P(X = Damage[Magnitude]) > P\ (X = Damage[Border])$$

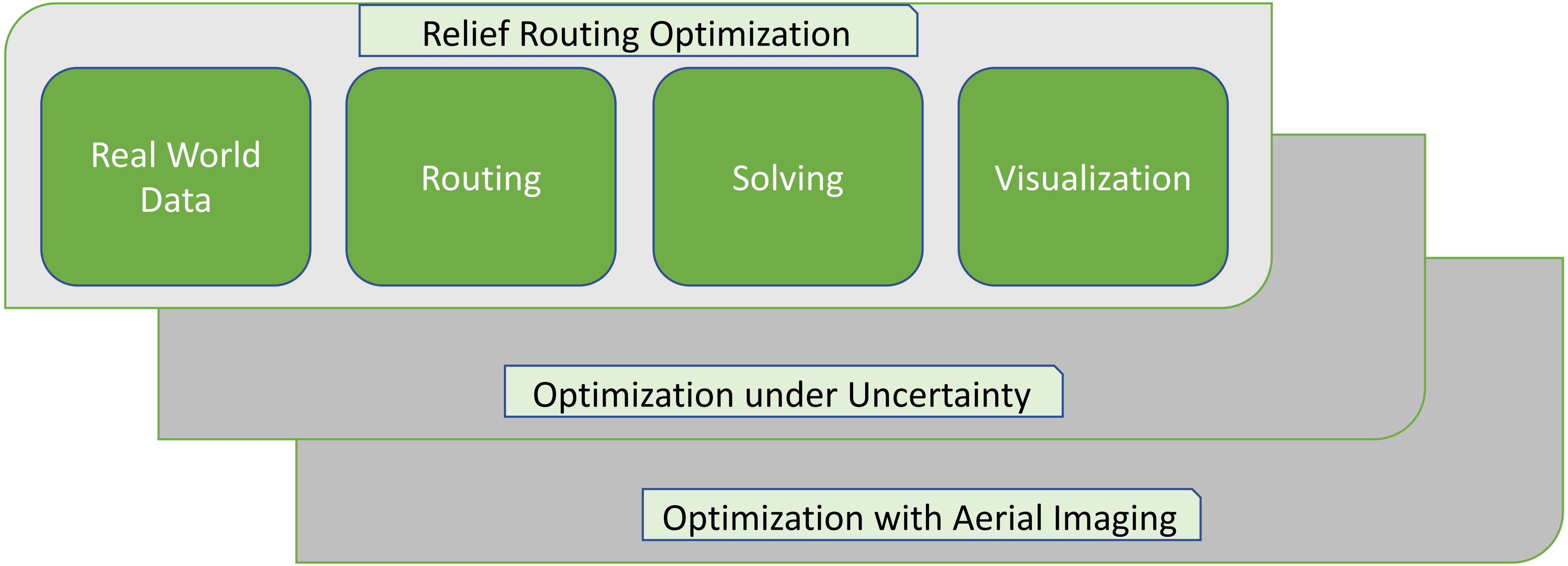
Research Question




Change Detection in Remote Sensing [im2]

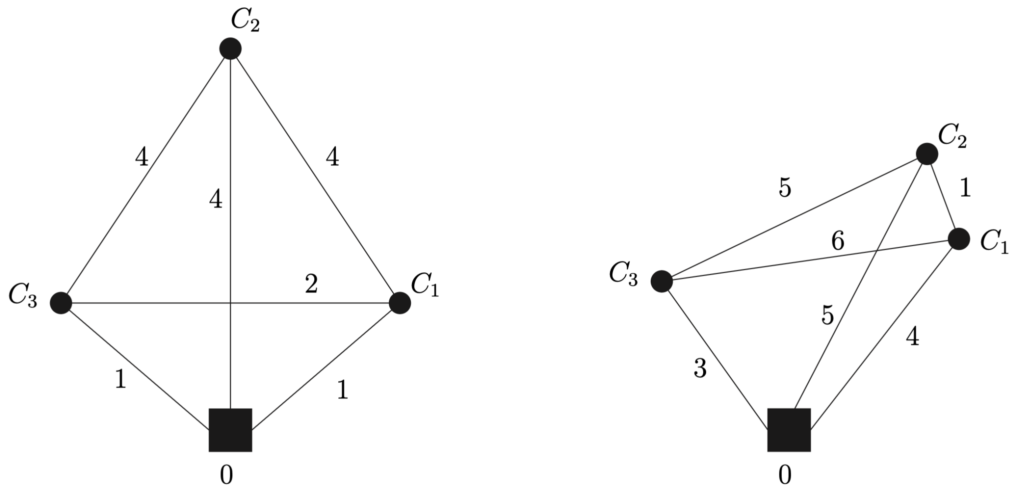
Research Question

Minimizing Response Time of Relief Vehicles



Method – MinMax, MinSum

- Network of Recipients
- Directed tours
- Objective functions for optimization
  - Minimization of the Maximum arrival time (MinMax)
  - Minimization of the Summation of arrival times (MinSum)



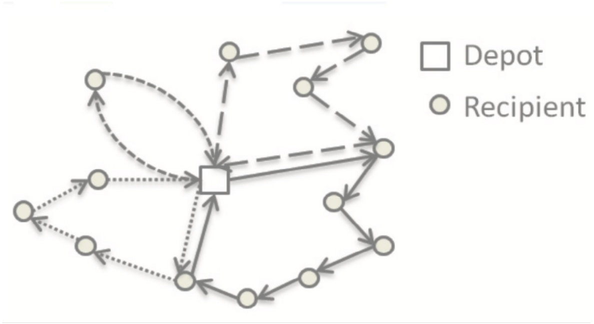
(a) TSP is different from minmax and minsum

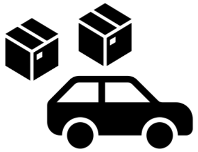
(b) minmax is different from minsum

[2]

Method – Theoretical Considerations

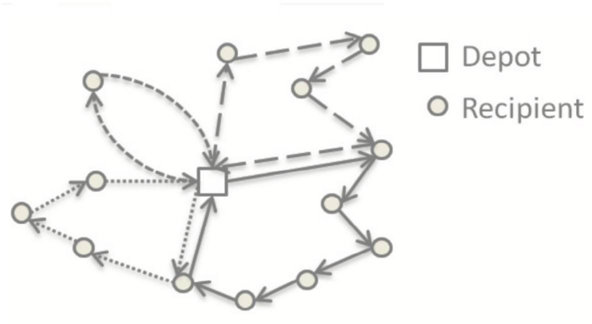
- Graph-network $G = (R, E, W)$
  - Recipients $R = \{r_1, \dots, r_m\}$
  - Edges $E = \{(e_{lk}) | e_{lk}\ connects\ r_l, r_k \in R,\ l \neq k\}$
  - Weights $W = \{w_{lk} | w_{lk}\ corresponds\ to\ e_{jk}\}$
- Capacity:
  - $\forall\ Vehicles\ n_i, \exists\ capacity\ c_i$



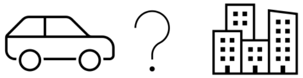


Method – Theoretical Considerations

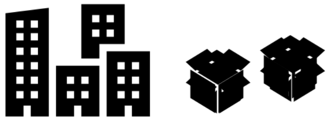
- Graph-network $G = (R, E, W)$
  - Recipients $R = \{r_1, \ldots, r_m\}$
  - Edges $E = \{(e_{lk}) | e_{lk} \; connects \; r_l, r_k \in R, \; l \neq k\}$
  - Weights $W = \{w_{lk} | \; w_{lk} \; corresponds \; to \; e_{jk}\}$
- Capacity:
  - $\forall \; Vehicles \; n_i, \exists \; capacity \; c_i$



+ Feasibility:
  - Vehicle n_i can reach recipient r_j

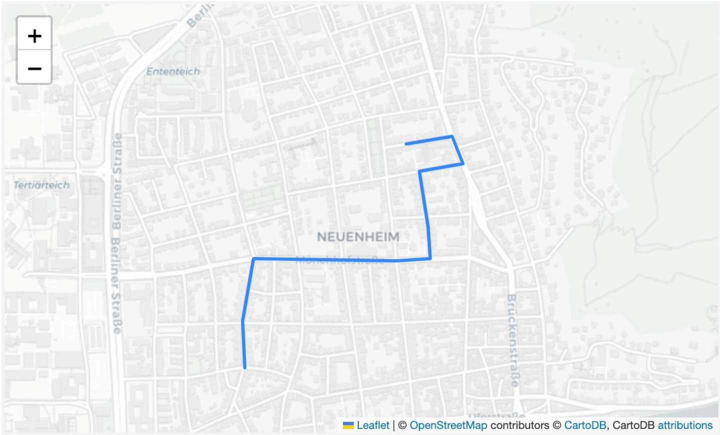


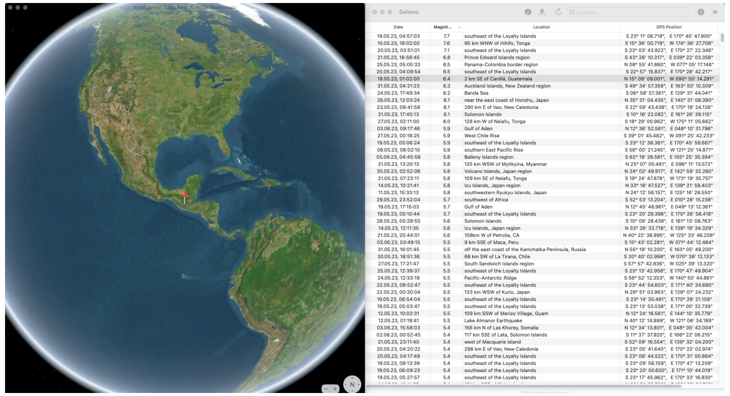
+ Demand:
  - $\forall \; Recipients \; r_j, \exists \; demand \; d_j$



+ Deadline:
  - $\forall \; Recipients \; r_j, \exists \; deadline \; t_j$

Method – Programmatical Implementation





https://github.com/python-visualization/folium

https://leafletjs.com/

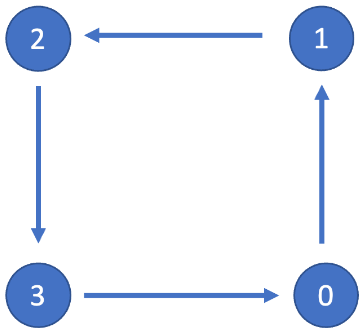
https://wiki.openstreetmap.org/wiki/API

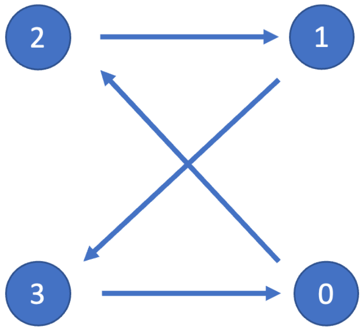
https://api.openrouteservice.org/

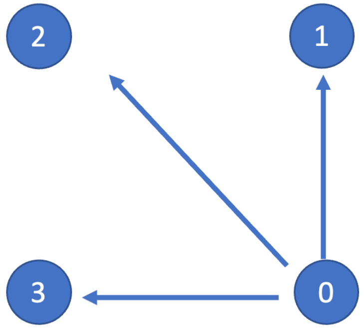
https://earthquake.usgs.gov/fdsnws/event/1/

```
|-API
|    |-Carthography
|    |-Routing
|    |-Seismic
|
|-Routing
|
|
|
|-Solver
|
|
|-Instanciation
|
|
|
```

Method – Programmatical Implementation

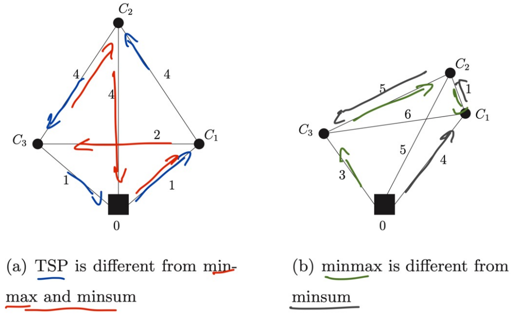







```
|-API
|    |-Carthography
|    |-Routing
|    |-Seismic
|
|-Routing
|    |-Directional
|    |-Permutational
|    |-Distantial
|
|-Solver
|
|
|-Instanciation
|
|
|
```

Method – Programmatical Implementation



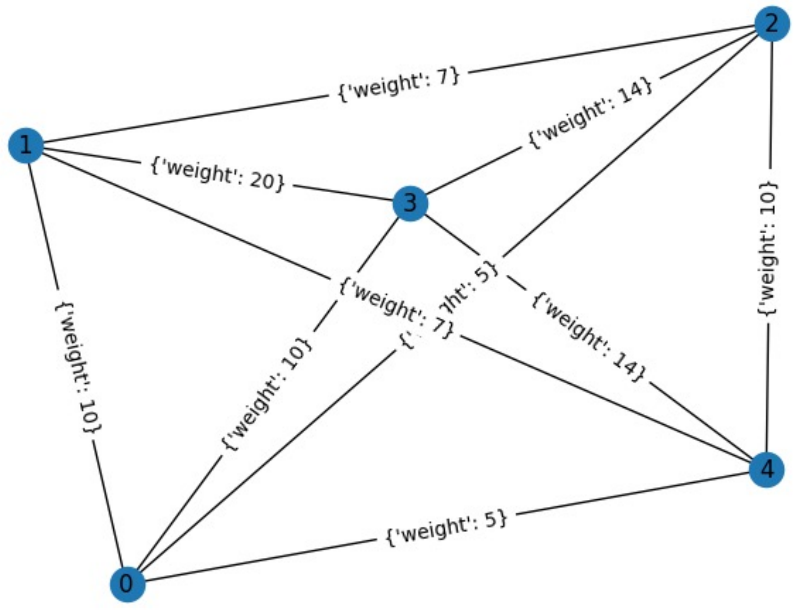
TSP, MinMax and MinSum with Notes [2]

- Integration of
  - Capacities
  - Feasibilities
  - Demands
  - Deadlines

```
|-API
|    |-Carthography
|    |-Routing
|    |-Seismic
|
|-Routing
|    |-Directional
|    |-Permutational
|    |-Distantial
|
|-Solver
|    |-MinSum
|    |-MinMax
|
|-Instanciation
|    |-Depot
|    |-Vehicle
|    |-Capacities
|
```

Experiments

- Theoretical Setting:
  - Graph with 1 depot and 4 recipients
  - Different distances
  - Uniform capacities

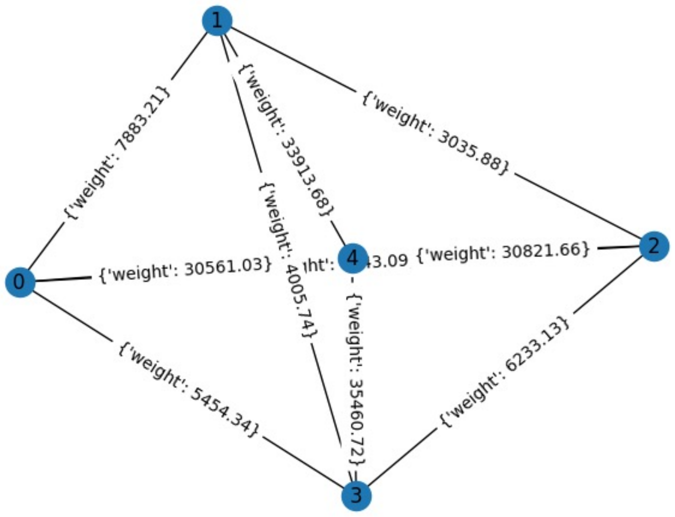


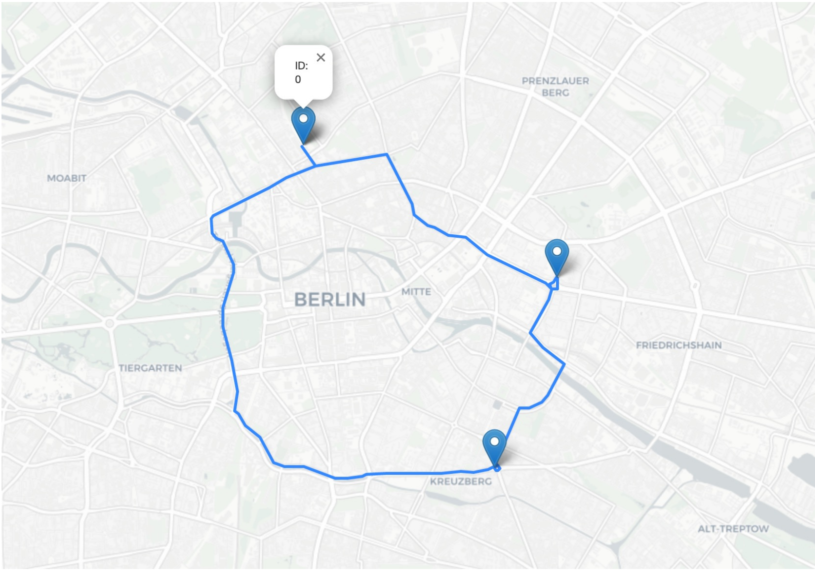
individual vehicle costs	Approach			
	Basic	**Feasibility**	**Deadline**	**Feasibility + Deadline**
MinMax	v1: 27, v2: 22	v1: 20, v2: 37	v1: 15, v2: 34	v1: 20, v2: 24
MinSum	v1: 27, v2: 22	v1: 20, v2: 24	v1: 29, v2: 27	v1: 29, v2: 27

Table 1. Individual vehicle costs for inexact implementations of the MinSum and MinMax

Experiments

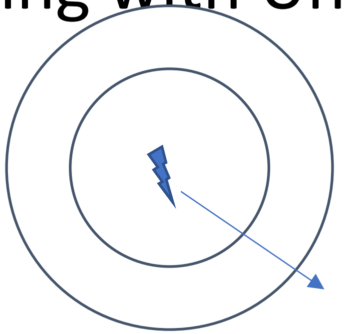
- Real-World Setting
  - Graph with 4 recipients
  - Real-world coordinates
  - Capacitated MinSum insertion algorithm
  - 2 vehicles



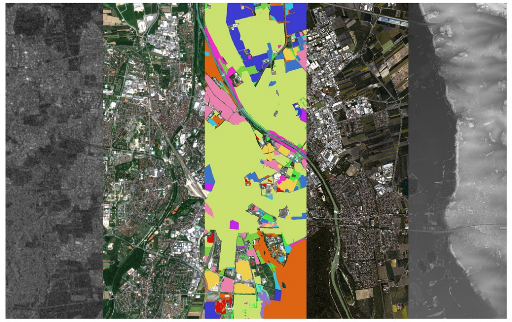


Conclusion

- Relief Routing with MinMax and MinSum Algorithms
- Integration of additional heuristics: Feasibilities, Demand, Deadlines
- Embedding of Cartography and Seismic activity with real-world and live data
    - End-to-End Integration: Instantiation → Visualization
- Future Work: Dealing with Uncertainties, Extensive Experiments



$$P(X = Damage[Magnitude]) > P\ (X = Damage[Border])$$



References

[1] M. Huang, K. Smilowitz, and B. Balcik, “Models for relief routing: equity, efficiency and efficacy,” Procedia-social and behavioral sciences, vol. 17, pp. 416–437, 2011.

[2] A. M. Campbell, D. Vandenbussche, and W. Hermann, “Routing for relief efforts,” Transportation science, vol. 42, no. 2, pp. 127–145, 2008.

[im1] Satellite image ©2023 Maxar Technologies Handout. (Reuters),

https://english.alarabiya.net/News/world/2023/02/09/Satellite-images-show-Turkey-before-and-after-tragic-quake-that-killed-over-16-000

[im2] Earth Systems Science Data – MDAS: a new multimodal benchmark dataset for remote sensing https://essd.copernicus.org/articles/15/113/2023/

Thank you

Contact: m.beese@campus.tu-berlin.de

https://github.com/bmarv/relief-routing-models

https://www.scienceofintelligence.de/people/marvin-beese/

Gefördert im Rahmen der Exzellenzstrategie von Bund und Ländern

Backup – Feasibilities & Deadlines

Algorithm 1 Pseudocode of the Minmax-Insertion Algorithm with Feasibilities and Deadlines

```
function MinMaxInsertion(distances, demands, capacites, deadlines, feasibilities)
    num_nodes, num_routes, routes, loads ← initiation from overloaded arguments
    sorted_nodes ← sorting in ascending order based on their deadlines
    for node in sorted_nodes do
        best_cost ← +∞
        best_vehicle ← −1
        best_position ← −1
        node_usability ← check wrt. feasibility, demand(node) and vehicle(loads)
        for vehicle in range(num_routes) do
            if vehicle ← feasible and deadline ← fulfillable and vehicle_capacity ← sufficient then
                current_cost ← calculate_route_cost
                insertion_cost ← cost_of_inserting(node, position, ∀ vehicles_on_route)
                if insertion_cost < best_cost then
                    best_cost ← insertion_cost
                    best_vehicle ← vehicle
                    best_position ← position
                end if
            end if
        end for
        insert node into routes[best_vehicle] at position best_position
        loads[best_vehicle] ← loads[best_vehicle] + demands[node]
    end for
    return routes
end function
```

Backup – Cost Algorithms

Algorithm 2 Cost Algorithms in Pseudocode

```
function CALCULATECOST(distance_matrix, route)
    cost ← 0
    for i from 0 to length(route) - 2 do
        from_node ← route[i]
        to_node ← route[i + 1]
        cost ← cost + distance_matrix[from_node][to_node]
    end for
    return cost
end function

function CALCULATEINSERTIONCOST(dist_mat, route, position, node)
    from_node assignment
    to_node assignment
    insertion_cost ← dist_mat[from_node][node] + dist_mat[node][to_node] - dist_mat[from_node][to_node]
    return insertion_cost
end function
```

Backup – TSP vs. MinMax or MinSum

- *la* := latest arrival time
- *sa* := sum of arrivals
- *us* := mean absolute upper semideviation [11]
  - $\frac{1}{n}\sum_{a_i \geq \mu}(a_i - \mu)$
  - smaller *us*: equity ↑
- *c* := total route duration

- results with shorter *la* & better *us*, but higher *c* for minmax compared to TSP
- results with shorter *la* & better *us*, but higher *c* for minsum compared to TSP

$$la(\frac{TSP}{MinMax}) > 1$$

$$sa(\frac{TSP}{MinSum}) > 1$$

		$\frac{la(TSP)}{la(MM)}$	$\frac{la(MS)}{la(MM)}$	$\frac{sa(TSP)}{sa(MS)}$	$\frac{sa(MM)}{sa(MS)}$	$\frac{c(MM)}{c(TSP)}$	$\frac{c(MS)}{c(TSP)}$	$\frac{us(TSP)}{us(MM)}$	$\frac{us(TSP)}{us(MS)}$	$\frac{us(MM)}{us(MS)}$
Augerat-A	MIN	0.948	0.970	0.974	0.973	1.004	1.027	0.926	0.902	0.850
	MAX	1.051	1.257	1.229	1.270	1.246	1.431	1.114	1.138	1.234
	AVG	1.013	1.114	1.089	1.124	1.075	1.184	1.016	1.016	1.003
Augerat-B	MIN	0.912	0.922	0.852	0.938	0.995	0.963	0.822	0.730	0.728
	MAX	1.146	1.275	1.424	1.321	1.233	1.384	1.391	1.565	1.315
	AVG	1.020	1.092	1.061	1.074	1.047	1.114	1.041	1.068	0.987
Golden	MIN	0.977	0.966	1.006	1.020	1.012	1.007	0.951	0.955	0.901
	MAX	1.049	1.263	1.185	1.210	1.095	1.303	1.089	1.080	1.052
	AVG	1.014	1.115	1.105	1.101	1.055	1.167	1.003	1.017	0.987

[2]

$$c(\frac{MinMax}{TSP}) > 1$$

$$c(\frac{MinSum}{TSP}) > 1$$

Backup – VRP vs TSP

- consideration of additional vehicles
- ratio gets significantly larger with multiple vehicles
  - $\frac{la(VRP)_Q^k}{la(MM)_Q^k}$
  - $\frac{sa(VRP)_Q^k}{sa(MS)_Q^k}$
- cost increases are not as significant

		$\frac{la(VRP)_Q^k}{la(MM)_Q^k}$			$\frac{c(MM)_Q^k}{c(VRP)_Q^k}$			$\frac{us(VRP)_Q^k}{us(MM)_Q^k}$		
		$k = 1$	$k = 5$	$k = 10$	$k = 1$	$k = 5$	$k = 10$	$k = 1$	$k = 5$	$k = 10$
Augerat-A	MIN	0.948	1.050	1.170	1.004	1.044	1.067	0.926	0.942	0.993
	MAX	1.051	1.618	1.627	1.246	1.220	1.370	1.114	1.376	1.305
	AVG	1.013	1.365	1.390	1.075	1.122	1.212	1.016	1.163	1.150
Augerat-B	MIN	0.912	0.890	1.066	0.995	1.013	0.994	0.822	0.622	0.771
	MAX	1.146	2.129	2.047	1.233	1.340	1.403	1.391	1.880	1.629
	AVG	1.020	1.390	1.390	1.047	1.129	1.180	1.041	1.180	1.126
Golden	MIN	0.977	1.032	1.233	1.012	0.972	0.925	0.951	0.968	1.086
	MAX	1.049	1.723	2.880	1.095	1.234	1.288	1.089	1.324	2.178
	AVG	1.014	1.364	1.683	1.055	1.120	1.138	1.003	1.141	1.340

[2]