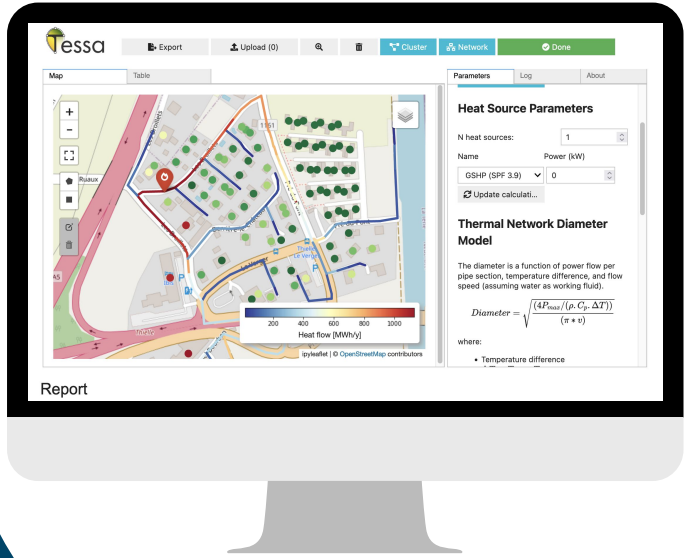
Planeto

Applying graph theory to clean energy districts

Energy Data Hack Days

Jonathan Chambers, CTO





Augmented MST Approach

Shortest Path - DOGE STICK RETRIEVAL

Weighted distance matrix

More experiments…

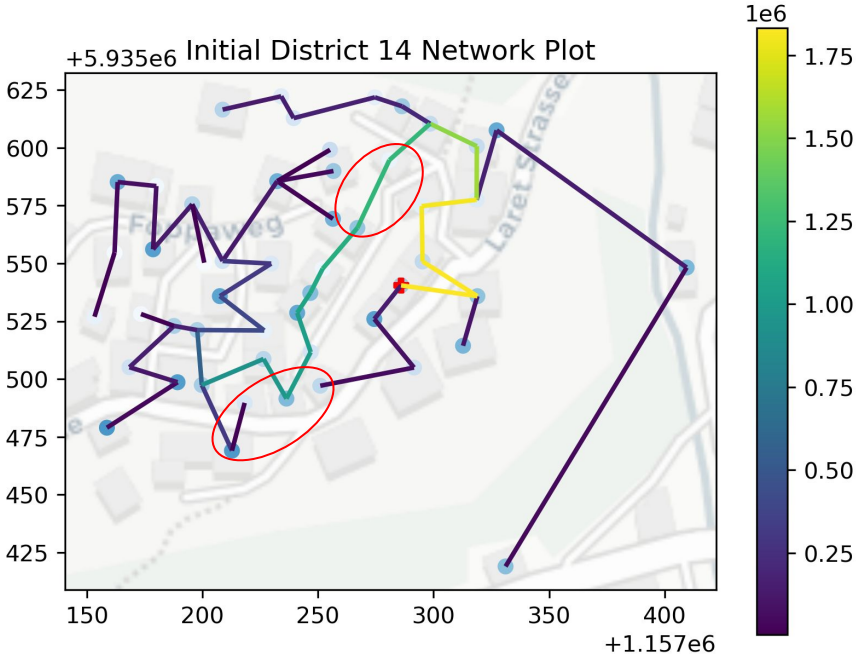
Cost Function

SUM(edge length * total energy flow through that edge)

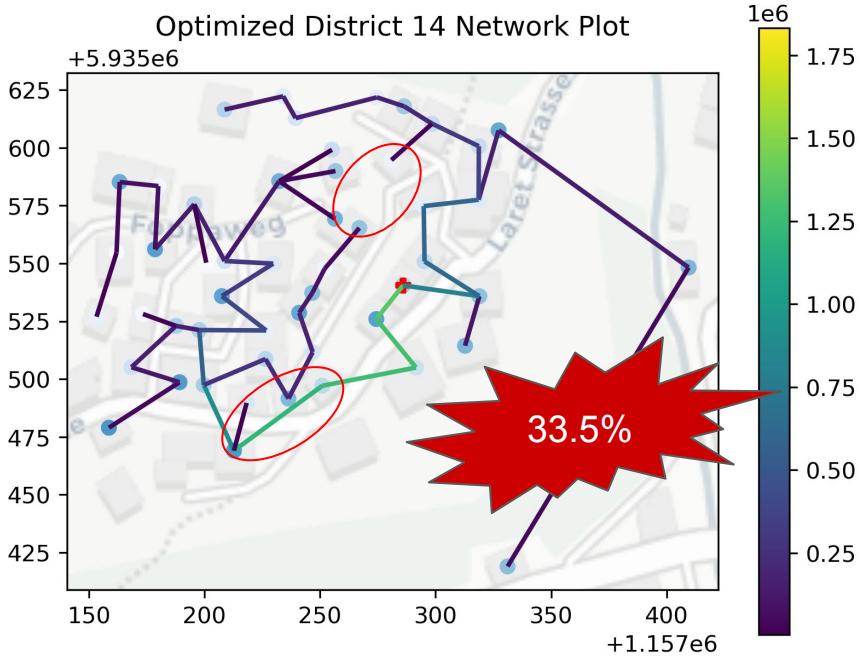
Augmented MST Approach

Intuition: randomly manipulate the edge length in the sparse input matrix until you achieve a lower cost function

Extension: penalize the edges proportional to their cost, as a batch and then one at a time until it improves the cost



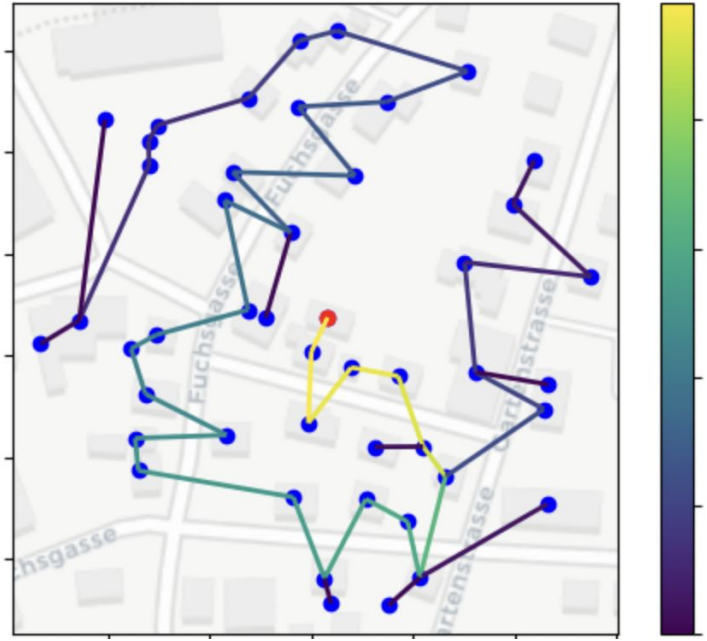
989 m



836 m…?

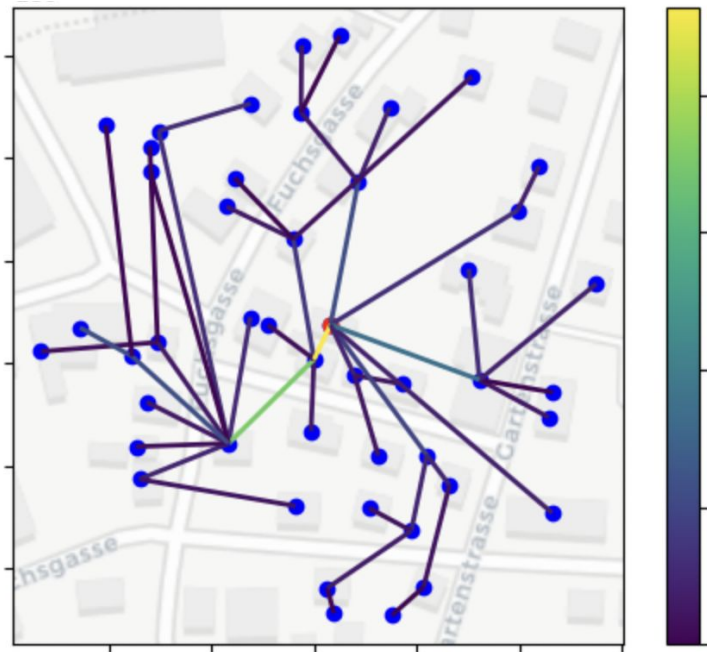
Taking the **shortest path to each consumer** leads to an average **cost reduction of 15%** and average length increase of 85%

Baseline

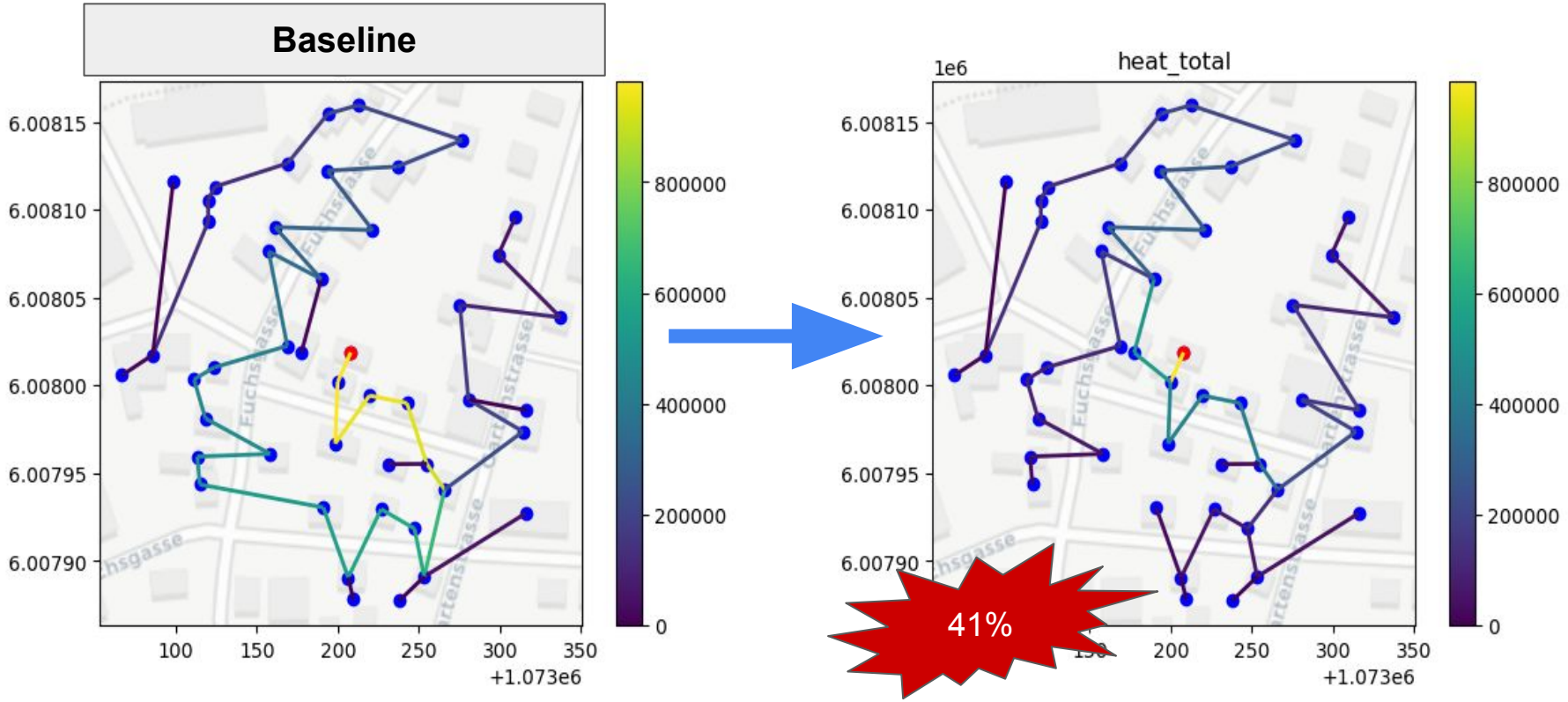


Optimized District, cost -56%, length +143%





MST using weighted distance matrix that balances both distances and consumption and finding optimal alpha [0,1]



Deterministic Approach

Goal: Adapt the distance of the edges according to the energy demands of the nodes such that MST finds better results according to the lost function.

Result: No significant improvements to the MST.

Reasons: The demand is too far away from the actuel lost calculation. The unknown capacity in this approach is maybe a to big downside.

Further steps: Improve the calculation of the adapted distances

Other Idea: Create all possible Graphs and search the best path according to the loss function. (Brute Force)

Other Algorithms

- Genetic Algorithm
  - Main Issue for this application: Random initialisation and crossover of individuals causes infeasible network flow
- Ant Colony Optimisation
  - Can be used as an alternative to MST for finding paths. Ants traverse the graph and follow pheromones of previous ants who give information on the goodness of their paths by leaving pheromones. Structure of paths run by ants is inherently different to the structure of a path made by water flow.
- Graph Neural Networks
  - Main Issue:
    - Problem formulation and loss function design (incorporating constraints)
    - Dependence of regression sub-tasks on each other
    - Incorporating edge characteristics into problem formulation
  - Most promising direction of further development:
    - Graph Attention Networks (Velickovic et. al, 2017)

Augmented MST performance

Random Perturbation

Case	% improvement	Case	% improvement
0	0%	10	1.1%
1	0%	11	0%
2	18.2%	12	0.2%
3	44.1%	13	1.4%
4	0%	14	27.8%
5	37.4%	15	15.8%
6	0%	16	0.2%
7	3.8%	17	0%
8	13.3%	18	0%
9	0%	19	0.9%

Penalty

Case	% improvement	Case	% improvement
0	8.8%	10	1.1%
1	4.6%	11	0%
2	18.2%	12	12.3%
3	54.3%	13	6.4%
4	10.5%	14	33.5%
5	37.4%	15	15.8%
6	2.8%	16	14.4%
7	3.8%	17	0%
8	39.0%	18	3.0%
9	0%	19	0.9%

Other Algorithms

- Genetic Algorithm
  - Algorithm: Initialise e.g. 500 “individuals” randomly which contain the parameters that are supposed to be optimised. Combine the best individuals and randomly mutate them to create a new generation. Repeat.
  - Main Issue for this application: Random initialisation and crossover of individuals causes infeasible network flow. The chances of finding a working configuration represents the main hurdle. GA should be possible, if the individuals can be initialised and kept feasible.
- Ant Colony Optimisation
  - Can be used as an alternative to MST for finding paths. Ants traverse the graph and follow pheromones of previous ants who give information on the goodness of their paths by leaving pheromones. Pheromone updates can be done based on loss function (path length AND capacity), however there exists an issue of early choices in a path being suboptimal in later stages. Another issue is that a single ant cannot “split” up at intersections like water could. This leads to the structure of paths run by ants being inherently different to the structure of a path made by water flow.
- Reinforcement Learning
  - Potential setup: Treat the network of roads as existing pipes with unknown capacity. The agent has #{number of pipes} continuous actions, i.e. it sets the size of the pipes. The reward function should penalise for unmet demand and penalise for infeasible nodes. Represent each node with its characteristics: {demand, inflow, outflow} where outflow = inflow - demand. Set the production of the substation as “negative” demand, such that it can be treated identically as all other nodes. To check if the configuration of the capacities is feasible, run a check on each node (outflow = inflow - demand) and attribute a variable penalty if this equation is not met. I believe this might work given that the assumption says supply == demand for the entire network.
  - (There was no time for this implementation, would require a full setup of OpenAI Gym environment + SB3/RLlib script for training).
- Linear Programming
  - We tried Linear Programming as well, but struggled to set the constraints such that the paths remain valid.
- Graph Neural Networks
  - Main Issue:
    - Problem formulation and loss function design (incorporating constraints)
    - Dependence of regression sub-tasks on each other
    - Incorporating edge characteristics into problem formulation
  - Most promising direction of further development:
    - Graph Attention Networks (Velickovic et. al, 2017)