OPTIMIZING SEARCH AND RESCUE PATHS IN THE ROCKY MOUNTAIN REGION

Matthew Knodell

Lillian Makhoul

Nicholas Rogers

AGENDA

- Motivation and Background
- Brief Dive Into Data Info and Cleaning
- Model Types
- Results
- Policy Recommendations

MOTIVATION

Approx. 3000 search and rescue incidents every year with 8000 hours spent searching.

Want to make sure these hours are spent searching the most important areas.

Our goal is to optimize the paths that rescue teams take while searching for subject(s).

We consider multiple starting locations and vary the number of teams out searching.

DATA INFORMATION

Data acquired from Mountain Rescue Association

Information we used from the data:

- X and Y coordinates of incident (latitude and longitude)
- Number of subjects

DATA CLEANING

Done using R

Sorted incidents by X and Y coordinates, then filtered out any that weren't in the Rocky Mountain Region.

Determined 'step size', or size of each region based on length and width of Rocky Mountain region and the number of nodes we wanted.

Assigned value to nodes based on the number of subjects found at that location.

Pseudocode:

1. Import CSV file
2. Filter to only include observations with region "Rocky Mountain Region"
3. Filter out observations with missing information on x and y coordinates
4. Remove outliers

CONSTRUCTING OUR DATA FILE

- Value was just a list of nodes and the corresponding number of subjects found
- Paths created using Python code (seen below)
- Necessary to print paths for each direction; arcs/paths are not bi-directional in AMPL

```python
import numpy as np

def print_neighbors_with_distance(matrix, max_distance):
    rows, cols = len(matrix), len(matrix[0])

    for i in range(rows):
        for j in range(cols):
            # Neighbors
            neighbors = []

            # Iterate over all possible neighbors within the max_distance
            for x in range(max(0, i - max_distance), min(i + max_distance + 1, rows)):
                for y in range(max(0, j - max_distance), min(j + max_distance + 1, cols)):
                    if (x, y) != (i, j) and abs(x - i) + abs(y - j) <= max_distance:
                        neighbors.append(matrix[x][y])

            # Print the result for the current element
            #print(f"At ({i},{j}): {matrix[i][j]}, Neighbors: {neighbors}")
            for k in neighbors:
                print(matrix[i][j],k)
```

DATA FILE

```
data;

set NODES := 1  2   3   4   5   6   7   8   9   10  11  12  13  14  15  16
# The number of nodes in NODES is based on the size of our grid

param start = 8; # we choose this
param stop = 25; # and this
param N = 12;    # and this

set paths := # The nodes that we can visit
    1 2      # Necessary to define two ways
    1 3      # 5 6 is different from 6 5
    1 9
    1 10
    1 17
    2 1
    2 3
```

```
param val := # Number of subjects found at each
    1   2    # node over the course of one year
    2   0
    3   3
    4   1
    5   3
    6   6
    7   0
    8   0
    9   0
    10  2
    11  5
    12  10
    13  7
```

MODEL TYPES: ASSIGNMENT AND SHORTEST PATH

- Assignment models are good for situations when we don’t have a strict time limit
- Shortest path models allow us to add in some element of “urgency” or a restriction on how much time the rescue teams have available.

WHY NOT TSP? ISN'T THIS THE PERFECT TIME FOR IT?

- Nope.
- We want to be able to start and stop at points that aren't necessarily the same.
- Subtours are a convenient way to determine how many teams we need to send out.
- Also, subtour elimination constraints are hard.

SHORTEST PATH MODEL

$$min \sum_{i,j} val_j \cdot X_{i,j} \quad \forall (i,j) \in Paths$$

$$s.t. \sum_{start,j} X_{start,j} = 1 \quad \forall j \in Paths$$

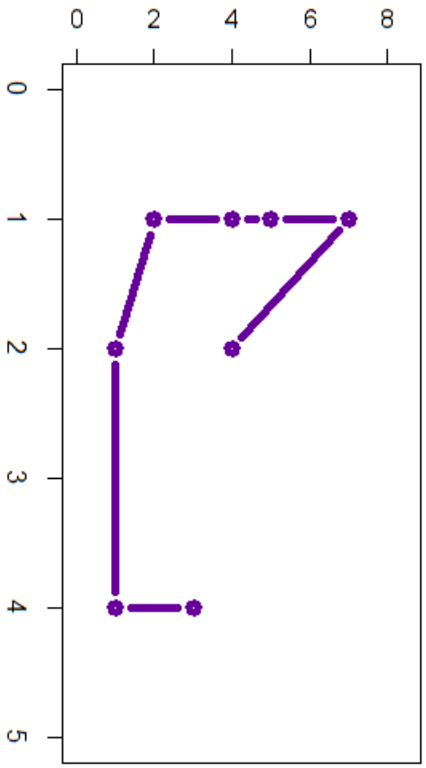
$$\sum_{j,stop} X_{j,stop} = 1 \quad \forall j \in Paths$$

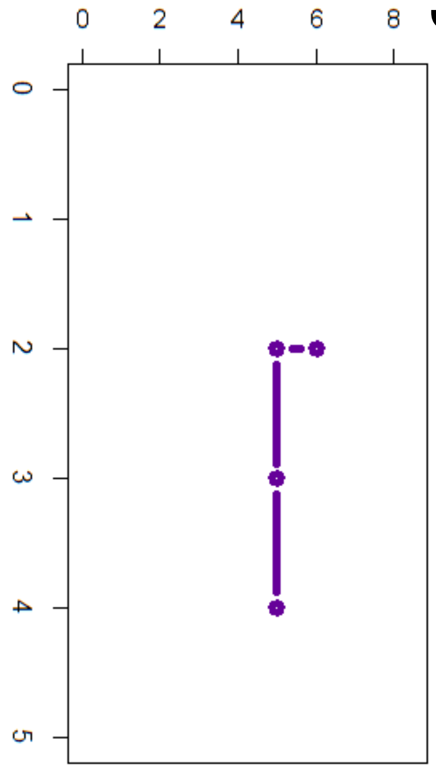
$$\sum_{i,j} X_{i,j} \leq N \quad \forall (i,j) \in Paths$$

$$\sum_{i,k} X_{i,k} = \sum_{k,j} X_{k,j} \quad \forall (i,k),(k,j) \in Paths,\ k \neq start, stop$$

$$X_{i,j} \geq 0 \quad \forall (i,j) \in Paths$$

SHORTEST PATH OUTPUT





WHAT IF WE GIVE IT AN INFEASIBLE SITUATION?

Surprise, it breaks.

We see negative flow going into some nodes, and flow greater than 1 going into others.

The following example uses a start of node 1 and a stop of node 50, but only allows for 2 steps.

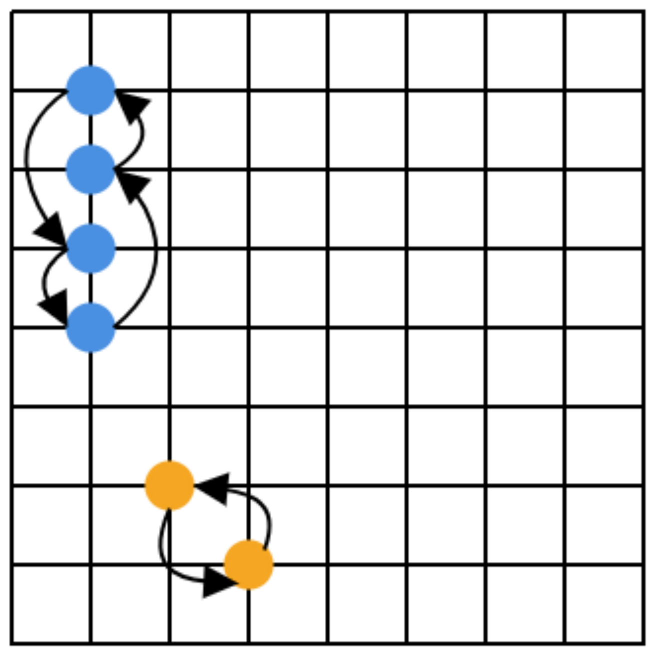
```
ampl: solve;
CPLEX 22.1.1.0: infeasible problem.
19 dual simplex iterations (0 in phase I)
constraint.dunbdd returned
19 extra dual simplex iterations for ray

suffix dunbdd OUT;
ampl: display X;
X [*,*]
:    1    2    3    4    5    6    7    8    9   10
1    .    0    2    .    .    .    .    .   -1    0
2    0    .    0    0    .    .    .    .    0    0
3    0    0    .    0    0    .    .    .    .    0
4    .    0    0    .    0    0    .    .    .    .
```

ASSIGNMENT MODEL

$$min \sum_{i,j} val_j \cdot X_{i,j} \quad \forall (i,j) \in Paths$$

$$\begin{aligned}
s.t. \ \sum_{j} X_{start,j} = 1, &\quad \forall (start, j) \in Paths \\
\sum_{i} X_{i,stop} = 1, &\quad \forall (i, stop) \in Paths \\
\sum_{i,k} X_{i,k} = \sum_{k,j} X_{k,j} &\quad \forall (i,k),(k,j) \in Paths \\
X_{i,j} \geq 0, &\quad \forall (i,j) \in Paths
\end{aligned}$$

ASSIGNMENT PROBLEM OUTPUTS



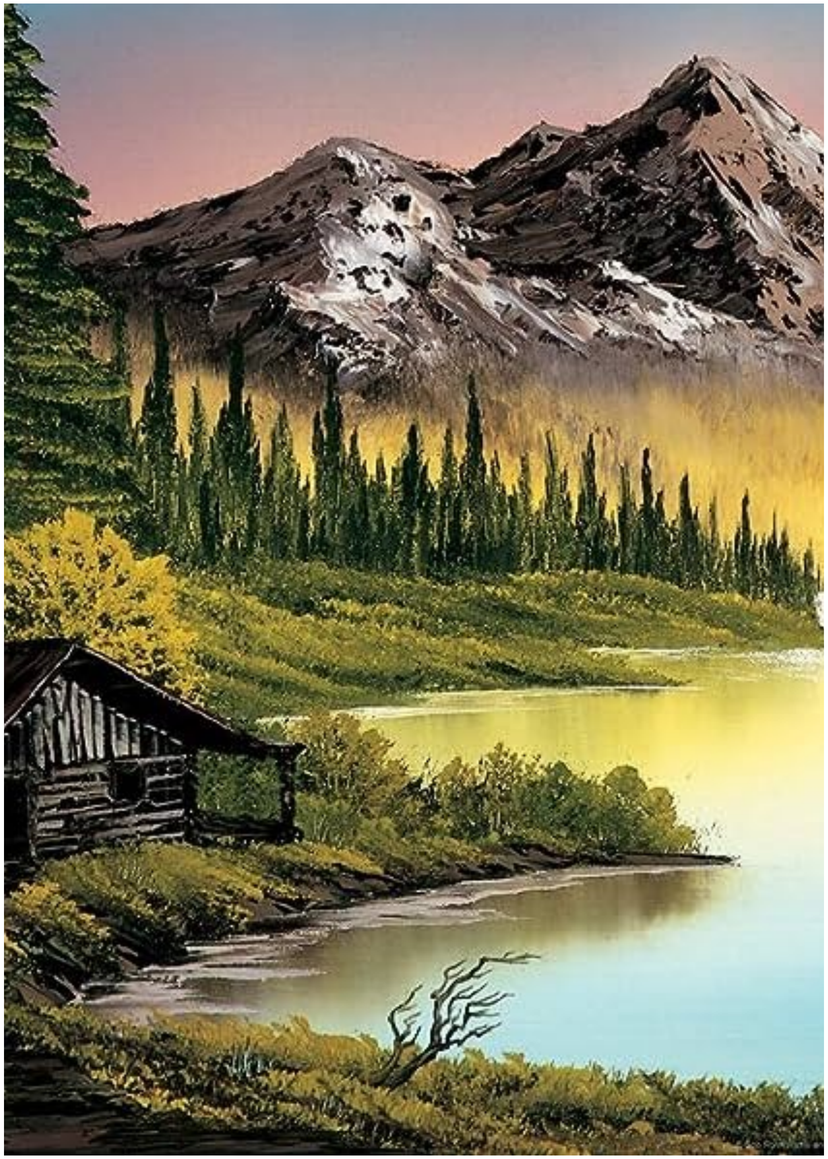
USES AND LIMITATIONS

Uses

- The shortest path model is ideal for time-sensitive situations.
- The assignment model is ideal for more time-flexible cases, where we want to search surrounding regions.

Limitations

- These paths are very general. In reality, it would vary on a case-to-case basis, specialized to the particular subject and any known history about them.
- We require a start and end point for this model, which may not always be provided.
- These models have no bearing on the geographic landscape, as the data available did not provide such.



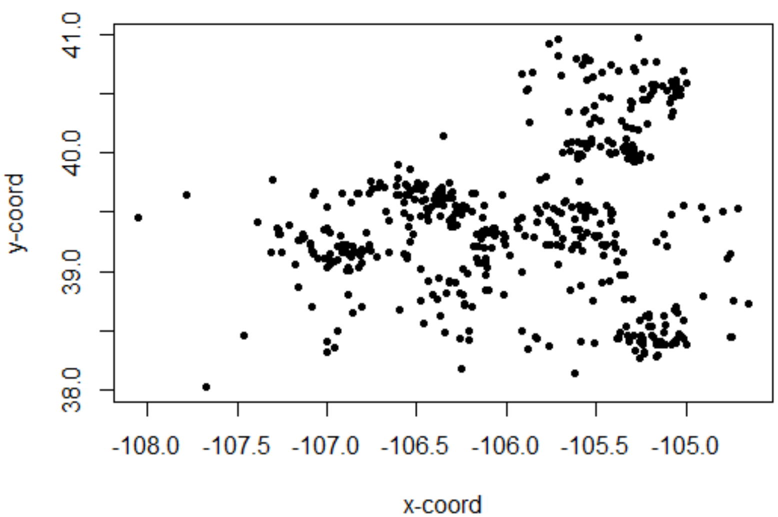
EXTENSIONS

- Further divide up subregions
- Split cases by type of incident (e.g. swift-water rescue, medical aid, recovery)
- Introduce multiple rescue teams from defined starting positions
- Provide additional constraints for different "types" of teams (aircraft, ground, etc.)

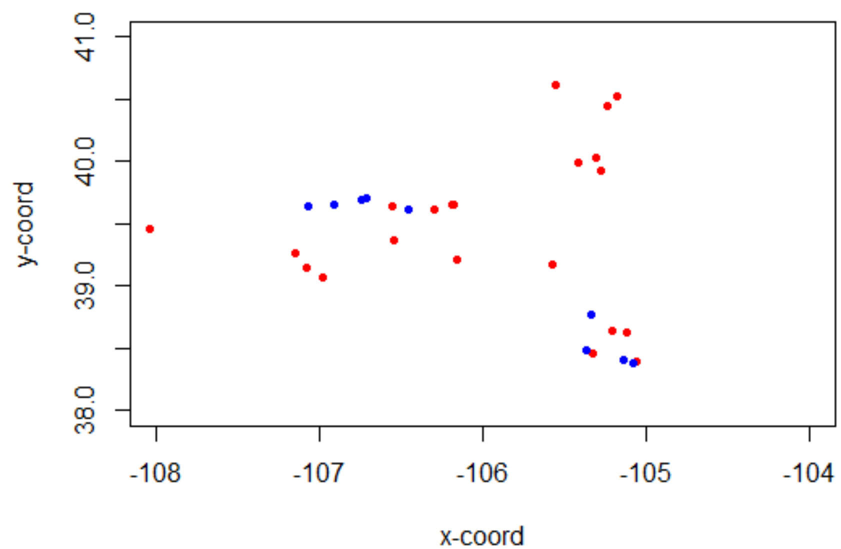
EXPANDING OUR MODEL TO A 75X75 GRID OF SUBREGIONS

- We are now looking at 5625 regions
- Each x-coordinate interval now varies by approximately 2.9 miles
- Each y-coordinate interval now varies by approximately 2.8 miles
- Each subregion now makes up approximately 8 square miles of the Rocky Mountain Region
- Results too large to visualize

SPLITTING UP OUR DATA BY TYPE OF INCIDENT



Scatter plot of each observation, plotted based on x and y coordinated of incident



Scatter plot of observations relating to recovery incidents (red), and observations relating to swift-water rescue incidents (blue)

THANK YOU!

Matthew Knodell
Lillian Makhoul
Nicholas Rogers