AMMA Consulting

Andrea Ira

Anna-Lena Hartinger

Mengyuan Zhang

Maharshi Sen

18.07.2024

Topics

1. Similarities

2. Algorithms: Dijkstra / A-Star

3. Computational Results and Analysis

4. Conclusion

5. Difficulties / Complications

Similarities

edge weight

*****

(1-percentage)

edge weight
*
(1-percentage)

First approach

- percentage: frequency of found edges in all historical routes

edge weight
*
(1-percentage)

First approach

- percentage: frequency of found edges in all historical routes

Second approach

- percentage: frequency of found edges in historical routes taken in one hour, i. e. from 00_00_00 to 00_59_59

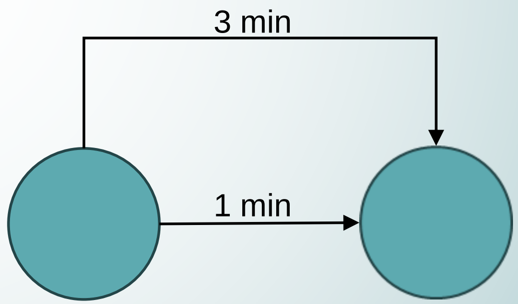
edge weight
*
(1-percentage)

First approach

- percentage: frequency of found edges in all historical routes

Second approach

- percentage: frequency of found edges in historical routes taken in one hour, i. e. from 00_00_00 to 00_59_59



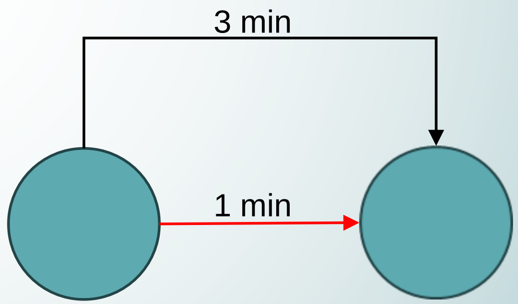
edge weight
*
(1-percentage)

First approach

- percentage: frequency of found edges in all historical routes

Second approach

- percentage: frequency of found edges in historical routes taken in one hour, i. e. from 00_00_00 to 00_59_59



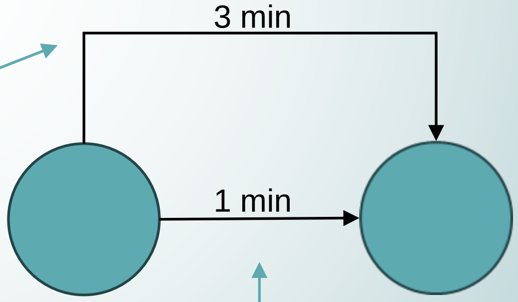
edge weight
*
(1-percentage)

First approach

- percentage: frequency of found edges in all historical routes

Second approach

- percentage: frequency of found edges in historical routes taken in one hour, i. e. from 00_00_00 to 00_59_59



percentage: 80%
→ new weight: 3 * (1-0.8) = 0.6

percentage: 10%
→ new weight: 1 * (1-0.1) = 0.9

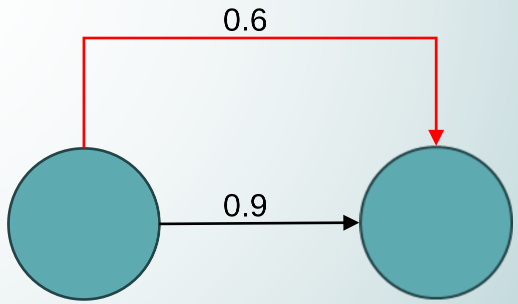
edge weight * (1-percentage)

First approach

- percentage: frequency of found edges in all historical routes

Second approach

- percentage: frequency of found edges in historical routes taken in one hour, i. e. from 00_00_00 to 00_59_59



→ new shortest, more optimal path

Algorithms

Dijkstra / A-Star

Dijkstra

Dijkstra

A*

Dijkstra

A*

edge weight: duration * similarity

Dijkstra

A*

edge weight: duration * similarity

Heuristic: euclidean distance from node 94 to node Y $\forall Y \in \{0, \dots, 537\}$

```python
def dijkstra_main(timestamp, per):
    city_graph2: WeightedGraph[str] = WeightedGraph([str(i) for i in range(538)])

    graph = pc.parse_csv()
    for i in range(1, len(graph)):
        fro = graph[i].get('from')
        to = graph[i].get('to')
        percentage = 1 - (per[i - 1])
        city_graph2.add_edge_by_vertices(str(fro), str(to),
            dur.get_edges_predicted_duration_new(timestamp)[i-1] * percentage)
```

Dijkstra

Implementation Similarity

```python
lst = []
for j in range(counter, len(p)-1):
    if fro == p[j+1].get("from"):
        percentage = 1 - per[j]
        lst.append((str(p[j+1].get("to")),
            dur.get_edges_predicted_duration_new(timestamp)[j] * percentage))
        if fro not in liste:
            liste.append(fro)
counter += 1
Graph_nodes[str(fro)] = lst
```

A*

Implementation Similarity

Dijkstra

A*

edge weight: duration * similarity

Heuristic: euclidean distance from node 94 to node Y $\forall Y \in \{0, \ldots, 537\}$

Final approach for optimal route with this similarity

$\rightarrow$ A*

Dijkstra

A*

edge weight: duration * similarity

Heuristic: euclidean distance from node 94 to node Y $\forall Y \in \{0, \ldots, 537\}$

Final approach for optimal route with this similarity

$\boxed{\rightarrow A^*}$

Why: is a better Dijkstra algorithm and returns the best optimal route out of all algorithms

Validating the similarity

Testing with the route-all.csv, validating with May instances from route-all-missing-last-day.csv

Validating the similarity

Testing with the route-all.csv, validating with May instances from route-all-missing-last-day.csv

First approach

The percentage of the routes, that are ≥75% similar to our optimal computed route, should lay between **5** and **10 %**

Validating the similarity

Testing with the route-all.csv, validating with May instances from route-all-missing-last-day.csv

First approach

The percentage of the routes, that are ≥75% similar to our optimal computed route, should lay between **5** and **10 %**

Second approach

The percentage of the routes, that are ≥75% similar to our optimal computed route, should lay between **10** and **30 %**

Validating the similarity

First approach

The percentage of the routes, that are ≥75% similar to our optimal computed route, should lay between **5** and **30 %**

Second approach

The percentage of the routes, that are ≥75% similar to our optimal computed route, should lay between **5** and **30 %**

Final approach:

Computational Results and Analysis

Results

Computational

- Mine
- Mengyuan
- Maharshi
- Final result

Mine

- A* optimal route
- Optimal in balance with similarity

Mengyuan

- 3 alternative routes to the computed optimal route
- As an addition to the optimal route
- Based on another similarity

Maharshi

- Dijkstra optimal route
- One alternativ route based on dtw-similarity
- Historical route

→ 3 routes the client can choose from

Final Result

- A* / Dijkstra optimal route
- 4 alternativ routes
- 1 historical routes

Analysis

Route

- Calculated route often also a historical route
- If not, then it is very similar to it

→ our route has with a high
kkpossibility a good balance
kkbetween optimal and similar

Similarity

- although the optimal route with similarities is longer than the shortest route, this is completely fine
- Balance between real life and theoretical world
- Shortest path algorithm without similarity is not applicable in real life

Conclusion

What you can expect from AMMA consulting group:

- Optimal routes everytime
- Additionally alternativ routes, similar to the optimal one
- Precise predictions

Difficulties / Complications

- Neural Network:

→ predicted speeds to high

→ balance

→ fine adjustment

→ accuracy

- Similarities:

→ adjustments to similarity approach

→ implement past similarities in the algorithm

→ right approach?

- How to improve:

→ over time, add more historical data to have a more accurate algorithm

→ more testing: is the similarity approach still working?

- Implementing and Coding

→ bad runtime: A* and Dijkstra compute 30 minutes for one route

- How to improve?

→ implement more efficiently, hire skilled developer

Difficulties

Thank You!

www.amma-consulting.com