TRAFFIC CONGESTION DETECTION WITH CROWDSOURCING DATA: A NETWORK-BASED SPATIAL-TEMPORAL CLUSTERING APPROACH

Zhihua (Jay) Zhang

8/1/2019

THE UNIVERSITY OF
TENNESSEE
KNOXVILLE

Outline

- Introduction
- Data and Methods
- Results
- Conclusion and Future work

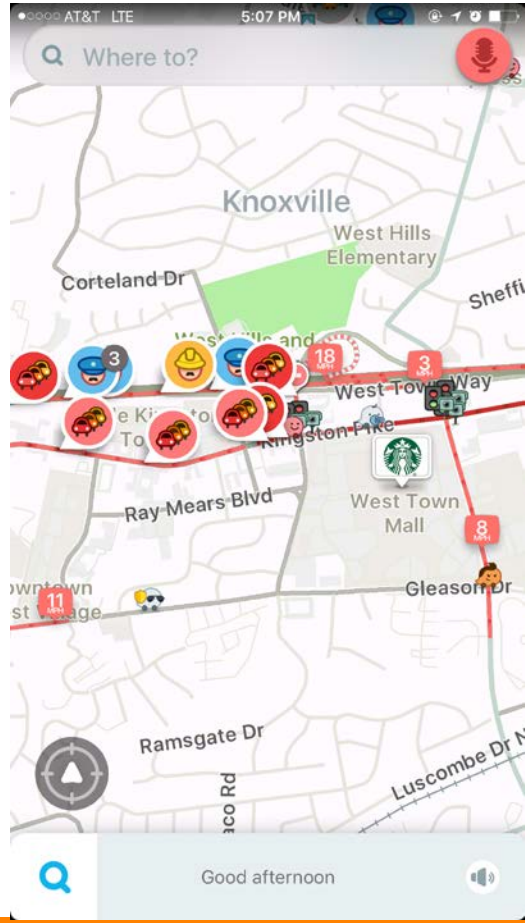
Introduction

Traffic Congestion

- Congestion detection, estimation and prediction
- Data limitation: missing value, limited coverage

Crowdsourcing data, like WAZE

- High coverage of the road network
- Cost-effective
- Reliable



develop and validate a network-based spatial-temporal clustering approach that supports the accurate detection of traffic congestion with Waze data

Data

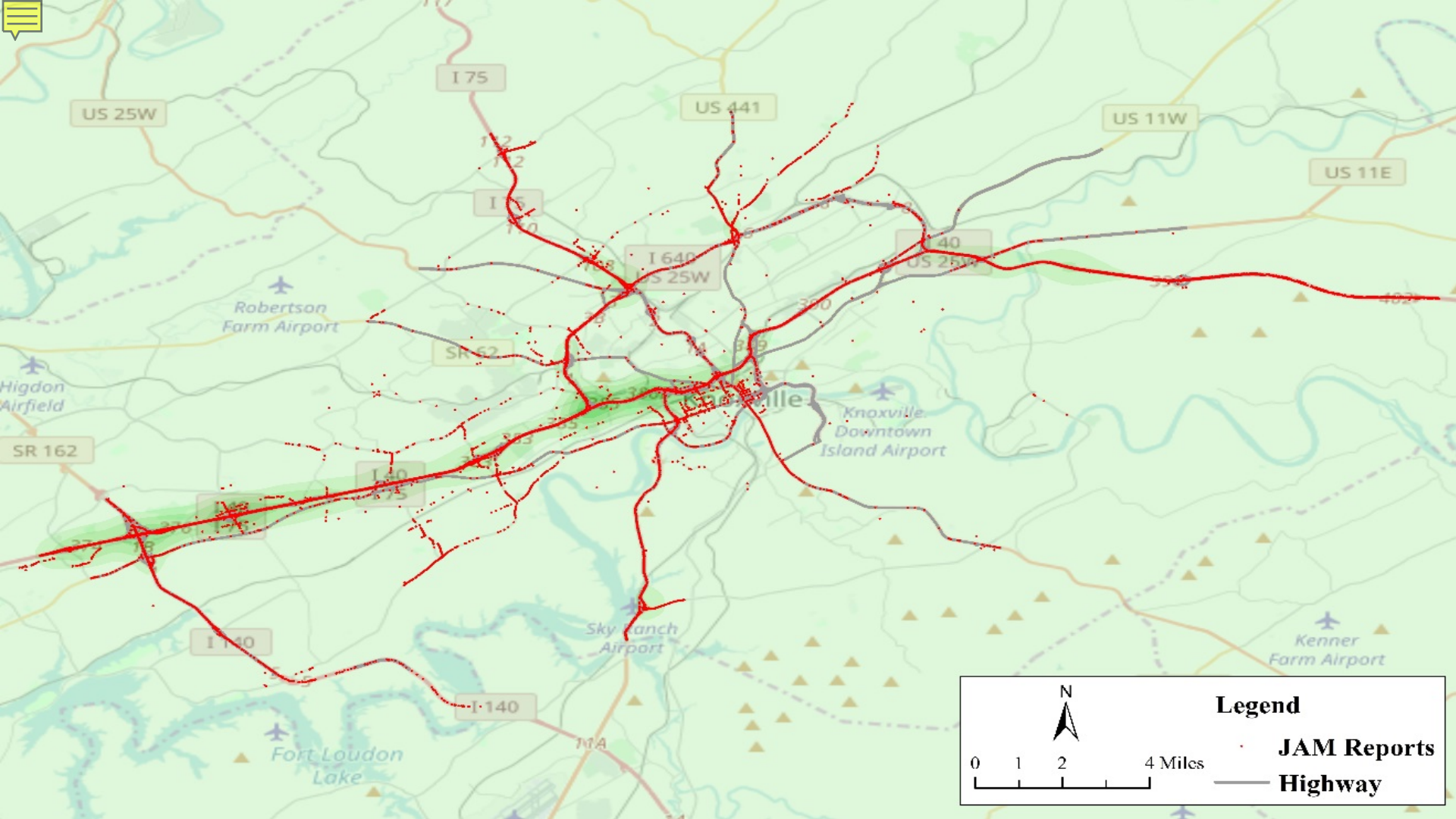
Road network (Open Street Map road network)

43,158 nodes

47,054 edges

Waze JAM reports for Knoxville (July-Sep, 2017)

17, 283 JAM reports


I 75
US 25W
US 441
US 11W
US 11E
I 640
US 25W
40
US 25W
Robertson
Farm Airport
SR 62
Higdon
Airfield
SR 162
I 40
Knoxville
Downtown
Island Airport
I 40
Sky Ranch
Airport
Kenner
Farm Airport
I 140
Fort Loudon
Lake
N
Legend
JAM Reports
Highway
0
1
2
4 Miles

Methods

Map matching algorithm

Project the data onto road

ST-DBSCAN

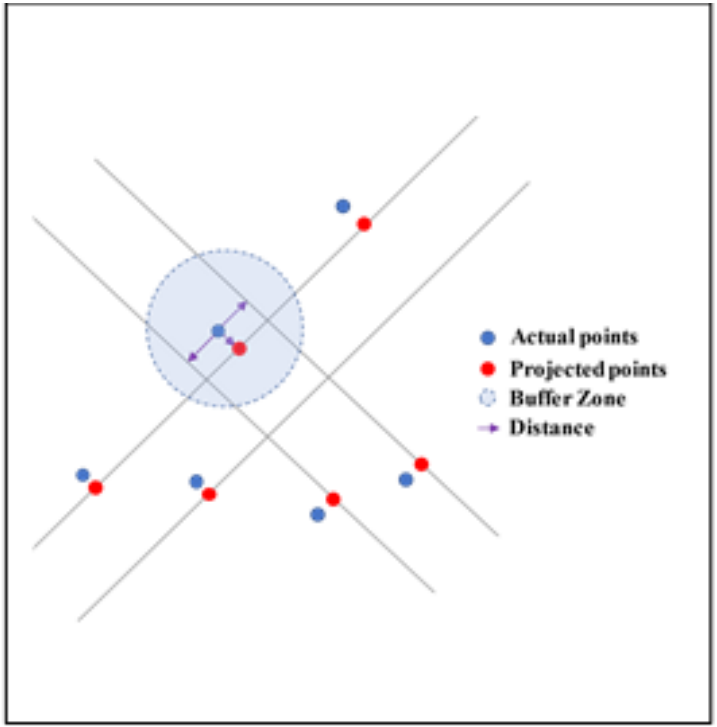
Density-based clustering

Dijkstra algorithm

Short-distance finding algorithm

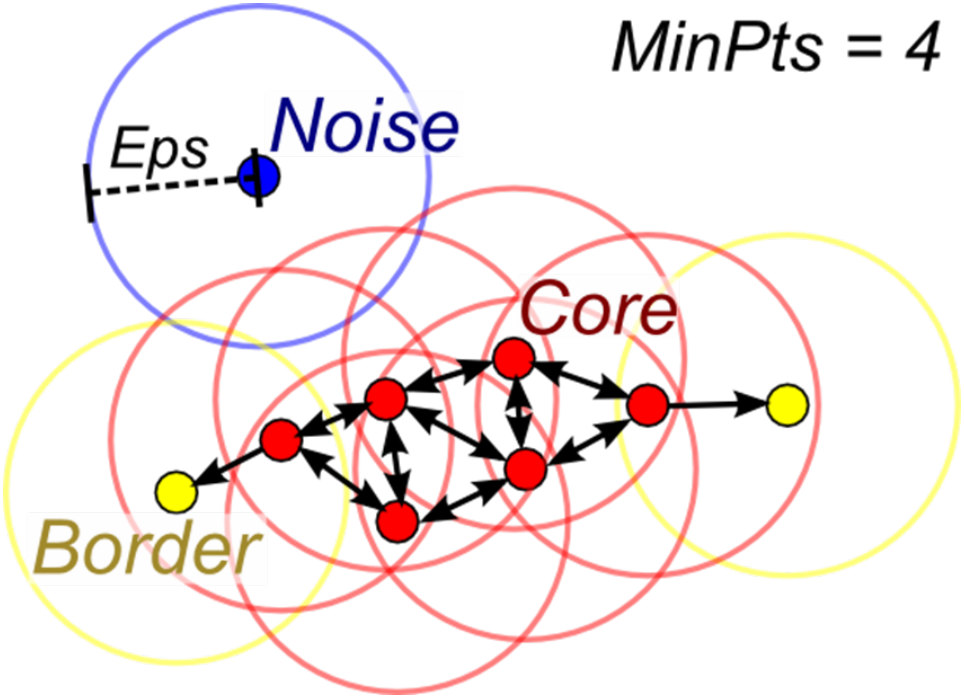
Map matching algorithm

```
MapMacthing (D, d)
    maxDist = infinity
    minDist = 0
    nearSegment = none
    for each point p in dataset D:
        create a buffer zone C with a distance d
        find the segments S intersects with C
        if len(C) == 0:
            continue
        for each segment s in S:
            if d[p, s] < maxDist:
                minDist = d[p, s]
                nearSegment = s
        project p onto the segment s
```



ST-DBSCAN

- Parameters
  - Distance threshold (ε)
  - Time threshold (t)
  - Minimum points ($minPts$)



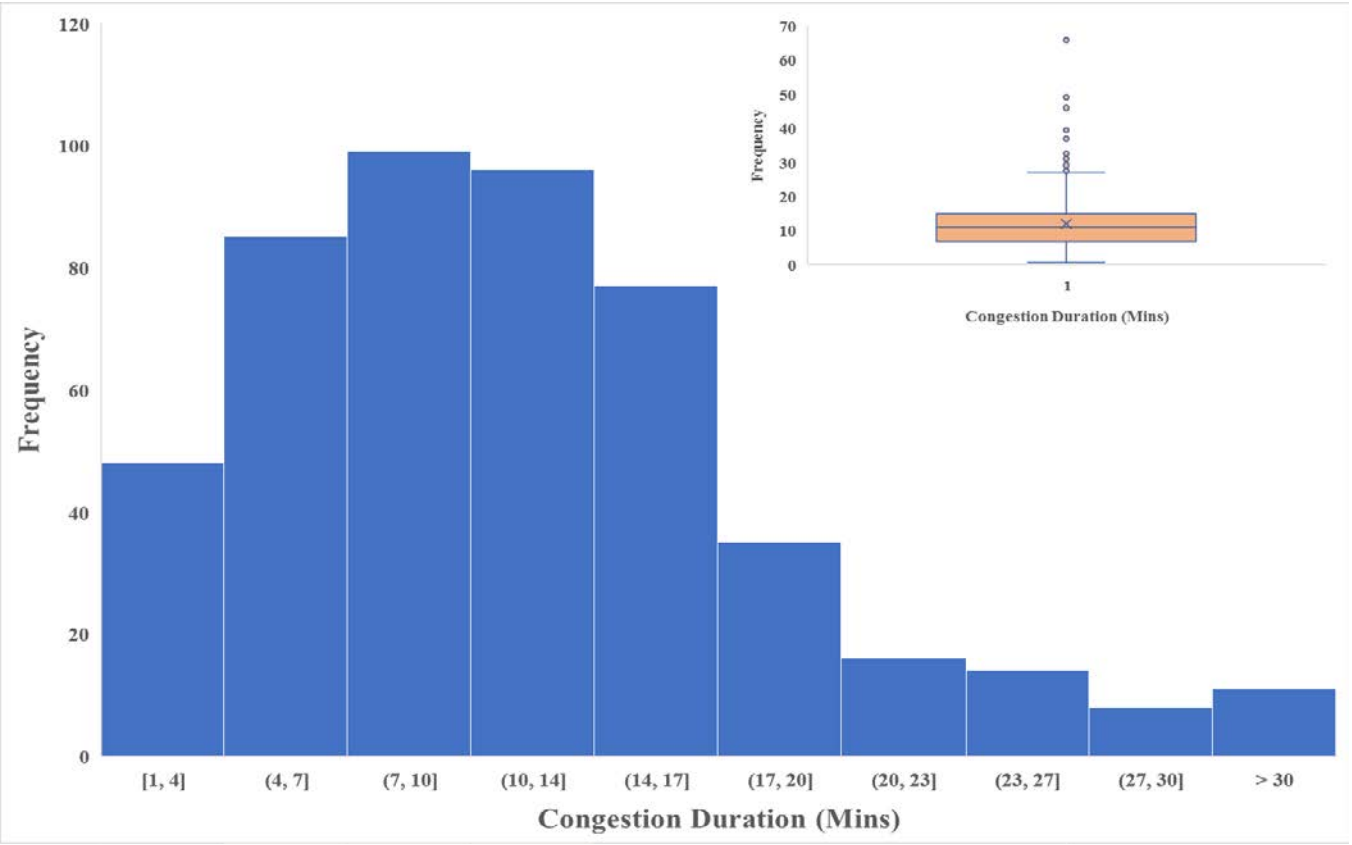
Dijkstra algorithm

```
Modified_Dijkstra(G, s, ε ){
        d[s] ← 0
        Q = {s}
        neighbor_list = {}
        while Q is not empty:
                u ←Q[0]
                Q←Q- {u}
                for each neighbor v of u:
                    if v is not visited and d[u]+e(u,v) < d[v]:
                        d[v] = d[u]+e(u,v)
                        Q←Q ⋃ {v}
            sort Q in ascending order based on the distance values of the elements
             if d[Q[0]] > ε :
                        return neighor_list
            else:
                        neighbor_list←neighbor_list ⋃ Q[0]
               mark Q[0] as visited
}
```

Distance threshold (ε)

- Get the neighbor points within distance threshold (ε)

Clustering Results

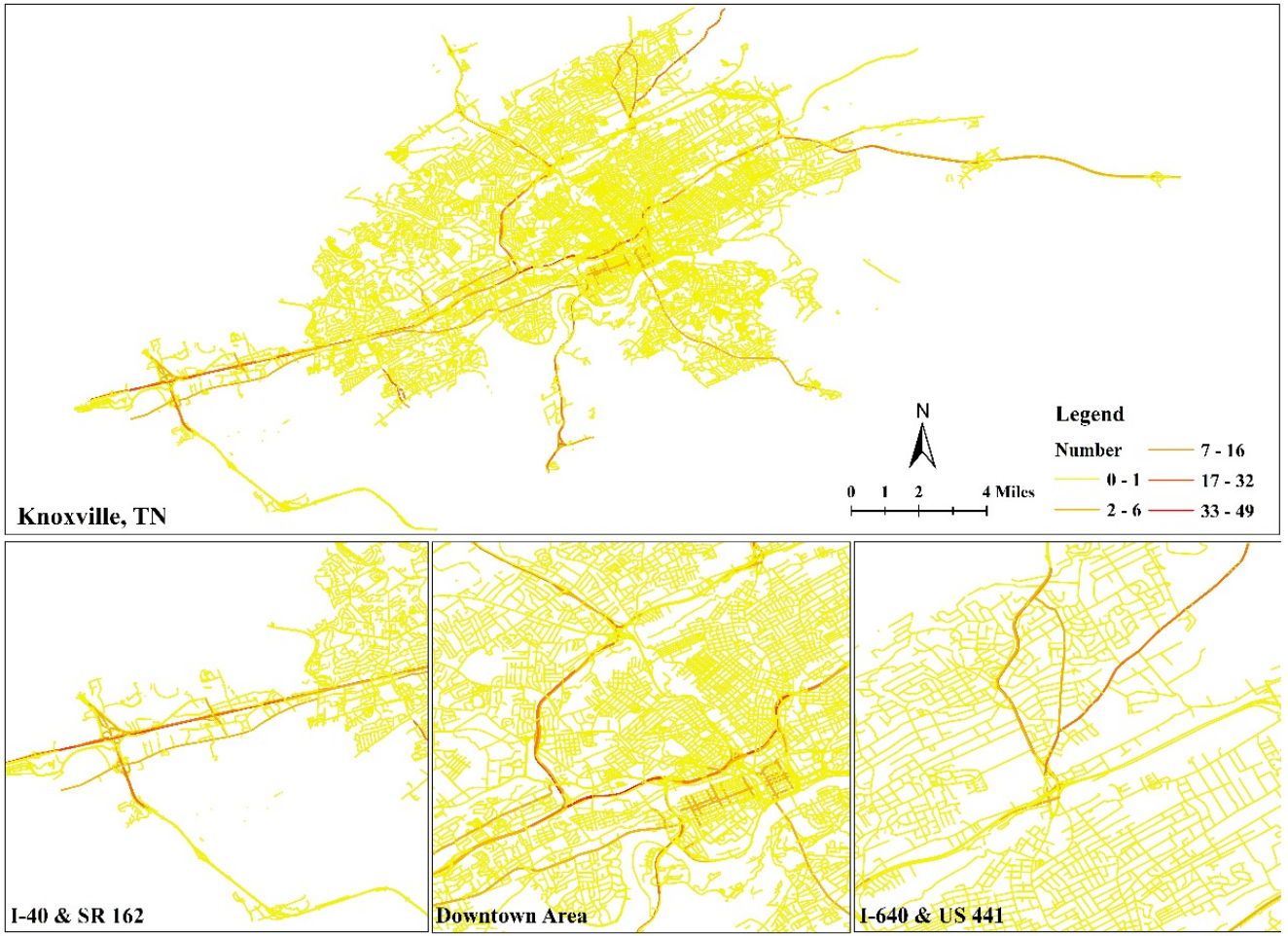


Congestion duration

- most of the traffic congestions are below 20 mins

Congestion level

- The congested road segments are mainly on interstate highways, especially at the highway interchanges



Conclusion and Future work

- Demonstrate the applicability of our algorithm to real world problems with a case study of Knoxville
- Our approach has the capability to identify cluster with any shape and a high routing flexibility.

- Big data application
- real-time application

Questions?

THE UNIVERSITY OF
TENNESSEE
KNOXVILLE