Georgia Tech College of Computing
Defining the New Face of Computing

DISL

Trajectory Clustering

Ling Liu
College of Computing
Georgia Institute of Technology

Introduction

- Pervasive use of GPS/Wifi-enabled devices
  - 295 million smartphones in 2010 to 1.2 billion units in 2015
- Explosion of location-based apps and services
  - LBS global revenue is from $2.8 billion in 2010 to $10.3 billion in 2015 [add citation]
- Huge amount of Location Data generated from multiple sources
  - GPS, WiFi, Social networks, RFID and senors




From Location Tracking to Location Analytics

- Increased interests and demand for deriving values and hidden insights from huge and growing location data
- Location Analytics (What):
  - Inference and knowledge discovery from mining location data of all types
- Opportunities
  - understanding how human mobility impacts the evolution of economy, transportation, healthcare and many aspects of the society we live and the infrastructure around us
- Broad LBS and apps:
  - Traffic management
  - Urban planning
  - Geo-marketing
  - Mobile adversting
  - Location-based recommendation systems

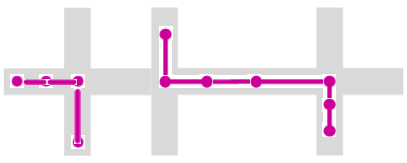
3

Three Types of Location Data

- Point based location data
  - Geometric position points from GPSs & sensors
- Area based location data
  - Symbolic address
  - A geometry area such as rectangle produced by WiFi localization, finger-print based localization, etc.

•Location analysis algorithms are cent ered on querying and mining point-based or area-based positions.

- Trajectory based location data
  - Mobile object trajectories, each consists of a timer series of position points, representing a travel path of a moving object



•Architectures & algorithms for analyzing and mining trajectories of mobile objects .

4

Trajectory Mining: Technical Challenges

- MO trajectories have complex characteristics
  - Temporal sequences of spatial location points
  - Varying-sample size sequences represent mobile objects' movement paths of varying lengths
    - Geometric points are unique and may contain outliers due to errors in location sensing and location acquisition
  - Moving object trajectories are typically constrained by the physical road network
- Large-scale trajectory analysis
  - High efficiency and high accuracy
  - Online/offline trajectory data processing and mining

Traditional mining algorithms fail to perform effectively on mobile object trajectory data

5

Research Hypothesis

- Trajectory mining should consider the physical paths of moving objects
  - road networks, walking paths
- Trajectory mining should take into account the motion behaviors and preserve the temporal sequence of trajectories.
- Trajectory mining should effectively handle varying length trajectories while minimizing any bias.
- Trajectory mining should consider semantic locations covered by mobile object trajectories
- Trajectory mining should efficiently target at both speed and quality, scalable in terms of volume, different types and sizes of trajectory data

6

Three Categories of Trajectory Mining Research

- Clustering location points of Trajectories
  - finding where in Atlanta is most crowded on Friday night
- Clustering whole trajectories
  - finding the traffic patterns in a city
- Clustering subtrajectories
  - finding the most frequently traveled road segments

7

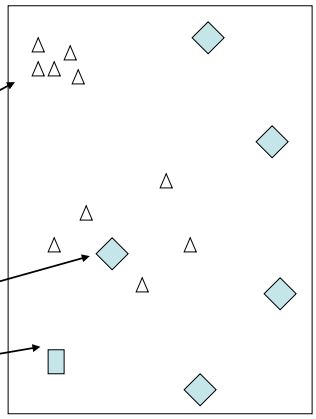
Clustering

- Definition: the process of grouping a set of physical or abstract objects into classes of *similar* objects [11]
- Applications: market research, pattern recognition, data analysis, image processing, etc.
- Representative algorithms: *k*-means [17], BIRCH [24], DBSCAN [6], OPTICS [2], and STING [22]
- Target data: previous research has mainly dealt with clustering of *point* data

8

Hot spots

- Question addressed
  - Is a phenomenon spatially clustered?
  - Which spatial entities or clusters are unusual?
  - Which spatial entities share common characteristics?
- Examples:
  - Cancer clusters [CDC] to launch investigations
  - Crime hot spots to plan police patrols
- Defining unusual
  - Comparison group:
    - neighborhood
    - entire population
  - Significance: probability of being unusual is high

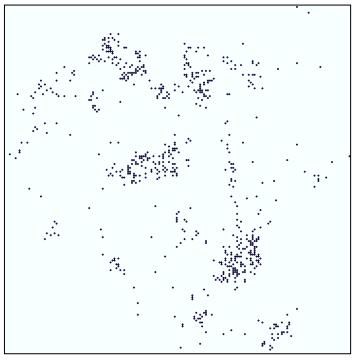


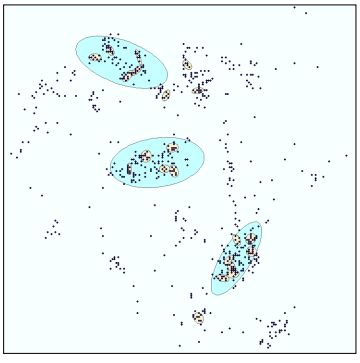
9

Spatial data exploration

- Hot spot analysis – Hierarchical NN

Cancer incidence data 1st and 2nd order clusters





- 10
- www.spatialanalysisonline.com
- 3rd edition

10

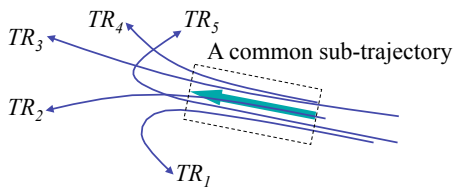
Algorithmic Ideas in Clustering

- Hierarchical –
  - All points in one clusters
  - then splits and merges till a stopping criterion is reached
- Partitional –
  - Start with random central points
  - assign points to nearest central point
  - update the central points
  - Approach with statistical rigor
- Density
  - Find clusters based on density of regions
- Grid-based –
  - Quantize the clustering space into finite number of cells
  - use thresholding to pick high density cells
  - merge neighboring cells to form clusters

11

Limitations of Existing Algorithms

- The position based clustering cannot provide insight on trajectories and trajectory patterns.
- The while trajectory clustering algorithm (such as those proposed by Gaffney et al. [7, 8]) clusters trajectories *as a whole.*
  - Clustering trajectories as a whole could not detect *similar portions* of the trajectories (i.e., common *sub*-trajectories)
  - **Example**: if we cluster TR_1~TR_5 as a whole, we cannot discover the common behavior since they move to totally different directions



12

Discovery of Common *Sub*-Trajectories

- Discovering common ***sub***-trajectories is very useful, especially if we have regions of special interest
  1) *Hurricane Landfall Forecasts* [18]
     Meteorologists will be interested in the common behaviors of hurricanes *near the coastline* or *at sea* (i.e., before landing)
  2) *Effects of Roads and Traffic on Animal Movements* [23]
     Zoologists will be interested in the common behaviors of animals *near the road* where the traffic rate has been varied
- Subtrajectory clustering solution is to partition a trajectory into a set of line segments and then group similar line segments
  ⇒ A ***partition-and-group*** framework
  ⇒ NEAT, a road network aware subtrajectory clustering framework

13

SIGMOD 2007

Trajectory Clustering: A Partition-and-Group Framework

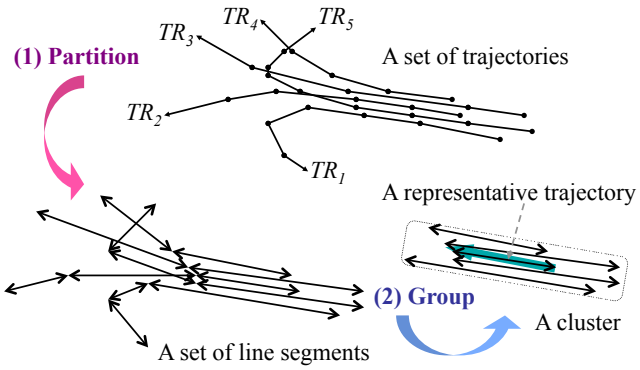
June 13, 2007

Jae-Gil Lee[1)], **Jiawei Han**[1)], **and Kyu-Young Whang**[2)]

[1)] Dept. of Computer Science, UIUC, USA
[2)] Dept. of Computer Science, KAIST, Korea

The *Partition-and-Group* Framework

- Consists of two phases: *partitioning* and *grouping*



Note: a representative trajectory is a common sub-trajectory

16

Problem Statement

- Given a set of trajectories $I = \{TR_1, \ldots, TR_n\}$, the TraClus algorithm generates a set of clusters $O = \{C_1, \ldots, C_m\}$ as well as a representative trajectory for each cluster C_i
- Necessary definitions:
  - A *trajectory* is a sequence of multi-dimensional points, which is denoted as $TR_i = p_1 p_2 p_3 \ldots p_j \ldots p_{leni}$
  - A *cluster* is a set of trajectory partitions; a *trajectory partition* is a line segment $p_i p_j$ ($i < j$), where p_i and p_j are the points chosen from the same trajectory
  - A *representative trajectory* is an imaginary trajectory that indicates the major behavior of the trajectory partitions

17

The Clustering Algorithm: *TRACLUS*

- Based on the partition-and-group framework

Algorithm **TRACLUS**
Input: A set of trajectories $I = \{TR_1, \ldots, TR_n\}$
Output: (1) A set of clusters $O = \{C_1, \ldots, C_m\}$
(2) A set of representative trajectories
Algorithm:
/* Partitioning Phase */
01: **for each** $TR \in I$ **do**
02: Partition TR into a set L of line segments;
03: Accumulate L into a set D;
/* Grouping Phase */
04: Group D into a set O of clusters;
05: **for each** $C \in O$ **do**
06: Generate a representative trajectory for C;

18

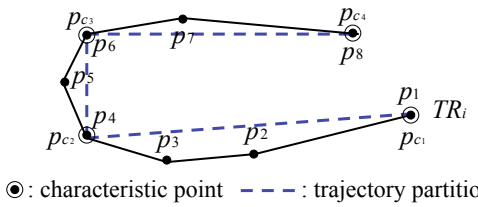
Current Step (1/3)

Algorithm **TRACLUS**
/* Partitioning Phase */
01: **for each** $TR \in I$ **do**
02: Partition TR into a set L of line segments;
03: Accumulate L into a set D;
/* Grouping Phase */
04: Group D into a set O of clusters;
05: **for each** $C \in O$ **do**
06: Generate a representative trajectory for C;

19

Characteristic Points

- Identify the points where the behavior of a trajectory changes rapidly; such points are called *characteristic points*

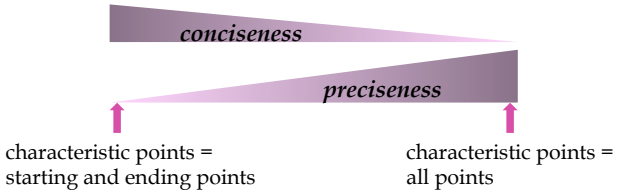


- A trajectory is partitioned at every characteristic point
- A line segment between consecutive characteristic points is called a *trajectory partition*

Desirable Properties of Trajectory Partitioning

- *Preciseness*: the difference between a trajectory and a set of its trajectory partitions should be as small as possible
- *Conciseness*: the number of trajectory partitions should be as small as possible

Note: two properties are contradictory to each other



⇒ We need to find the optimal tradeoff

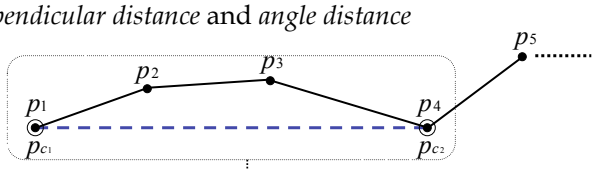
Minimum Description Length (MDL) Principle

- The MDL principle has been widely used in information theory
- The MDL cost consists of two components [9]: ***L(H)*** and ***L(D | H)***, where H means the hypothesis, and D the data
  - $L(H)$ is the length, in bits, of the description of the hypothesis
  - $L(D \mid H)$ is the length, in bits, of the description of the data when encoded with the help of the hypothesis
- The best hypothesis H to explain D is the one that minimizes the sum of $L(H)$ and $L(D \mid H)$

22

Translation into MDL Optimization

- Finding the optimal partitioning translates to finding the best hypothesis *using the MDL principle*
  - $H \Rightarrow$ a set of trajectory partitions, $D \Rightarrow$ a trajectory
  - $L(H) \Rightarrow$ the sum of the length of all trajectory partitions
  - $L(D \mid H) \Rightarrow$ the sum of the difference between a trajectory and a set of its trajectory partitions, computed by the combination of the *perpendicular distance* and *angle distance*



$$L(H) = \log_2(len(p_1p_4))$$
$$L(D \mid H) = \log_2(d_\perp(p_1p_4, p_1p_2) + d_\perp(p_1p_4, p_2p_3) + d_\perp(p_1p_4, p_3p_4)) + \log_2(d_\theta(p_1p_4, p_1p_2) + d_\theta(p_1p_4, p_2p_3) + d_\theta(p_1p_4, p_3p_4))$$

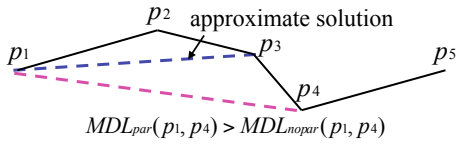
- $L(H)$ measures *conciseness*; $L(D \mid H)$ *preciseness*

23

Approximate Trajectory Partitioning

- The cost of finding the optimal partitioning is prohibitive
- Use an approximate algorithm; our approximation is to regard the set of local optima as the global optimum
- Algorithm skeleton (see detail in the paper – Fig. 8):
  - Compute the MDL costs for the case when a point p_k is a characteristic point and the case when p_k is not

$$\begin{cases} \text{Choose } p_{k-1} \text{ as a characteristic point,} & \textit{if the former} > \textit{the latter} \\ \text{Advance } p_k \text{ by increasing } k, & \textit{otherwise} \end{cases}$$



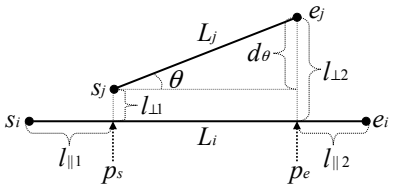
Current Step (2/3)

Algorithm **TRACLUS**
/* Partitioning Phase */
01: **for each** $TR \in I$ **do**
02: Partition TR into a set L of line segments;
03: Accumulate L into a set D;
/* Grouping Phase */
04: Group D into a set O of clusters;
05: **for each** $C \in O$ **do**
06: Generate a representative trajectory for C;

Distance between Line Segments

- The weighted sum of three components: the *perpendicular distance*($d_\perp$), *parallel distance*($d_\parallel$), and *angle distance*(d_θ)
  - Adapted from similarity measures used in the domain of pattern recognition [4]

$$dist(L_i, L_j) = w_\perp d_\perp + w_\parallel d_\parallel + w_\theta d_\theta$$



$$d_\perp = \frac{l_{\perp 1}^2 + l_{\perp 2}^2}{l_{\perp 1} + l_{\perp 2}}$$
$$d_\parallel = \mathrm{MIN}(l_{\parallel 1}, l_{\parallel 2})$$
$$d_\theta = \|L_j\| \times \sin(\theta)$$

Remark: the sum of the distances between endpoints does not work well for line segment clustering

26

Density of Line Segments

- Change the density definitions for points, originally proposed for DBSCAN [6], to those for line segments
- **Def.** (*ε-neighborhood*):

  $N_\varepsilon(L_i) = \{L_j \in D \mid dist(L_i, L_j) \le \varepsilon\}$
- **Def.** (*core line segment*):

  L_i is a core line segment w.r.t. ε and *MinLns* if $|N_\varepsilon(L_i)| \ge MinLns$
- **Def.** (*directly density-reachable*):

  L_i directly density-reachable from L_j w.r.t. ε and *MinLns* if $L_i \in N_\varepsilon(L_j)$ and $|N_\varepsilon(L_j)| \ge MinLns$
- **Def.** (*density-reachable*):

  Transitive closure of directly density-reachability
- **Def.** (*density-connected set* $\equiv$ *cluster*):

  1) Maximal w.r.t. density-reachability

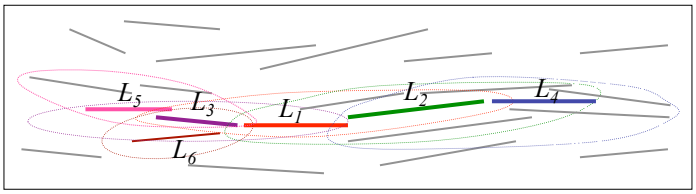
  2) Any line segments are density-connected, i.e., density-reachable from a third line segment

27

Density of Line Segments (cont'd)

- **Example**:
  - L_1, L_2, L_3, L_4, and L_5 are core line segments
  - L_2 (or L_3) is directly density-reachable from L_1
  - L_6 is density-reachable from L_1, but not vice versa
  - L_1, L_4, and L_5 are all density-connected

MinLns = 3

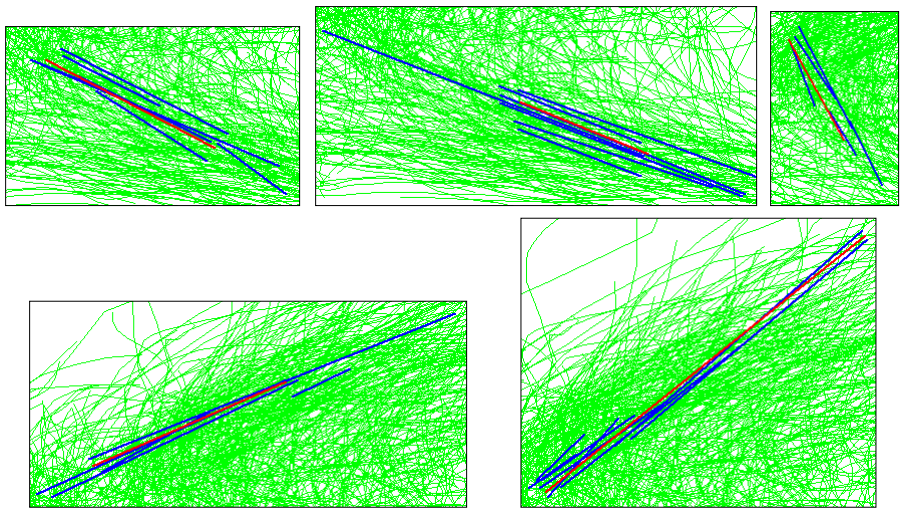


L_6 $L_5 \longleftrightarrow L_3 \longleftrightarrow L_1 \longleftrightarrow L_2 \longleftrightarrow L_4$

Note: the shape of an ε-neighborhood is likely to be an ellipse rather than a circle

28

Examples of ε-neighborhoods



Red lines: core line segments, **Blue lines**: line segments in the ε-neighborhood

29

Line Segment Clustering

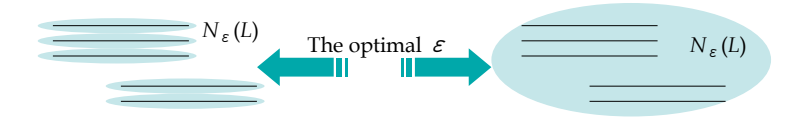
- Algorithm skeleton (See Fig. 8 in the paper):
  1. Select an unprocessed line segment L
  2. Retrieve all line segments ***density-reachable*** from L w.r.t. ε and *MinLns*
     - If L is a core line segment ($|N_\varepsilon(L)| \geq MinLns$), a cluster is formed
     - Otherwise, L is marked as a noise
  3. Continue this process until all line segments have been processed
  4. Filter out clusters whose trajectory partitions have been extracted from too few trajectories

- Time complexity (See Lemma 3 in the paper):
  - $O(n^2)$: if an index does not exist
  - $O(n \log n)$: if an index does exist

30

Heuristic for Parameter Value Selection

- **Estimation of ε**
  - Find the value of ε that minimizes the ***entropy*** of $|N_\varepsilon(L)|$ (ε-*neighborhood*)
    - Good clustering: $|N_\varepsilon(L)|$ tends to be skewed ⇒ the entropy is small
    - Worst clustering: $|N_\varepsilon(L)|$ tends to be uniform ⇒ the entropy is large



Too small ε → every $|N_\varepsilon(L)| = 1$

Too large ε → every $|N_\varepsilon(L)|$ = # of line segments

- **Estimation of *MinLns***
  - Choose one from $avg(|N_\varepsilon(L)|) + 1$, default MinLns ~ 3
    - *MinLns* should be larger than $avg(|N_\varepsilon(L)|)$ to discover meaningful clusters

31

Current Step (3/3)

Algorithm **TRACLUS**
/* Partitioning Phase */
01: **for each** *TR* ∈ *I* **do**
02: Partition *TR* into a set *L* of line segments;
03: Accumulate *L* into a set *D*;
/* Grouping Phase */
04: Group *D* into a set *O* of clusters;
05: **for each** *C* ∈ *O* **do**
06: Generate a representative trajectory for *C*;

32

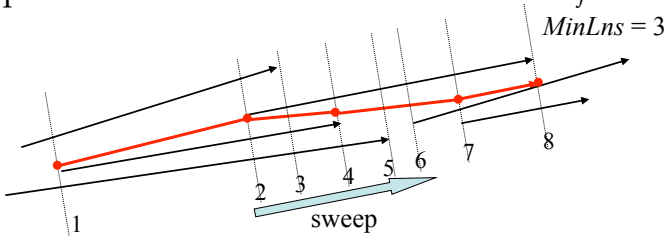
Representative Trajectories

- Describe the overall movement of the trajectory partitions that belong to the cluster
- Correspond to common sub-trajectories
- Useful for domain experts to understand the movement in the trajectories

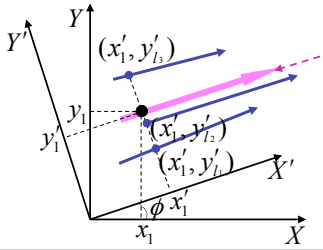
33

Representative Trajectory Generation

- Sweep a vertical line in the direction of the *major axis*



- Compute the average w.r.t. the *average direction vector*



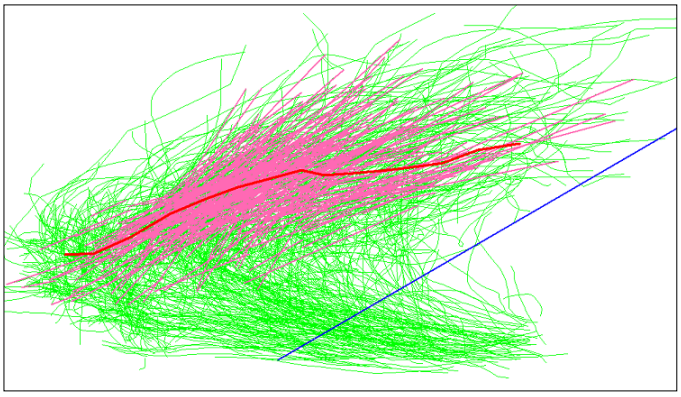
average direction vector

average coordinate
in the $X'Y'$ coordinate system

$$(x_1', y_1') = (x_1', \frac{y_{l_1}' + y_{l_2}' + y_{l_3}'}{3})$$

34

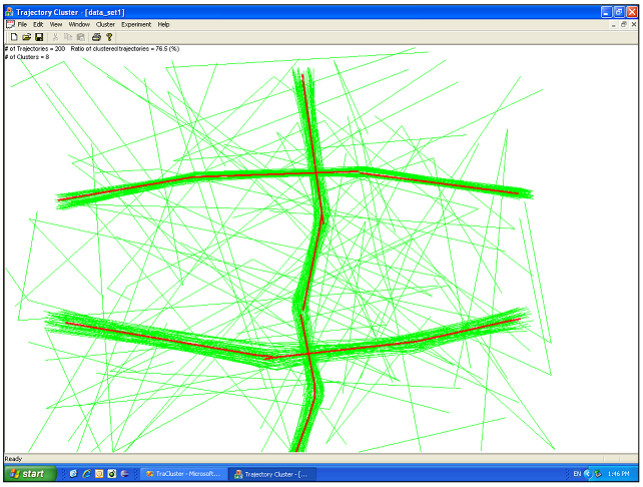
An Example of a Representative Trajectory



A red line: a representative trajectory,
A blue line: an average direction vector,
Pink lines: line segments in a density-connected set

35

A Quick View of a Clustering Result



Simple synthetic data: 200 trajectories (25% are noises)

36

Performance Evaluation

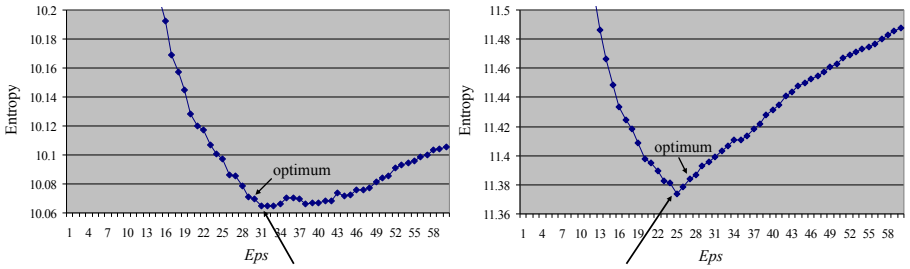
- Use two real trajectory data sets
  - Hurricane track data set
    - Record the Atlantic hurricanes from the years 1950 through 2004
    - Contain 570 trajectories and 17,736 points
  - Animal movement data set
    - Record the locations of elk, deer, and cattle from the years 1993 through 1996 (the Starkey project)
    - *Elk1993*: Contain 33 trajectories and 47,204 points;
      Deer1995: Contain 32 trajectories and 20,065 points

- Validate the clustering quality
  1) Estimate the parameter values for ε and *MinLns*
  2) Try a few values around the estimated ones; determine the optimal parameter values by visual inspection

37

Effectiveness of Parameter Estimation

- Entropies depending on the value of ε



ε with the minimum entropy: an estimated value

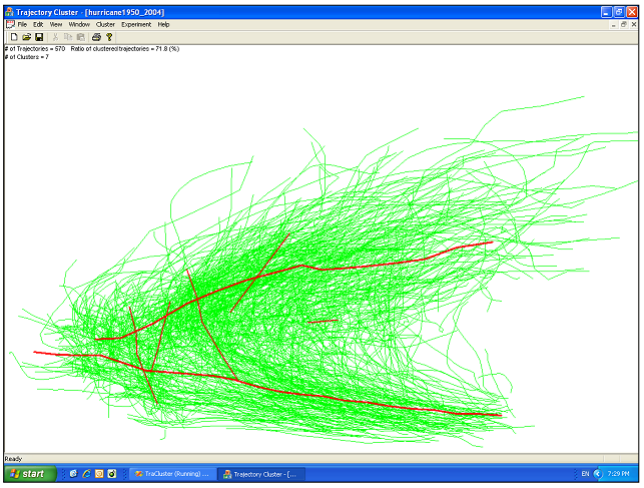
(a) Hurricane Tracks (b) *Elk1993*

The optimal value is very close to the estimated value

⇒ The accuracy of the heuristic is quite high

38

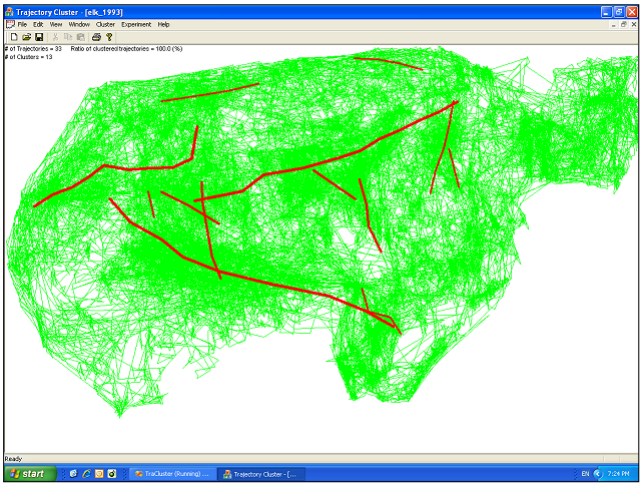
Clustering Result: Hurricane Tracks



ε = 30 and *MinLns* = 6 → # of clusters = 7

39

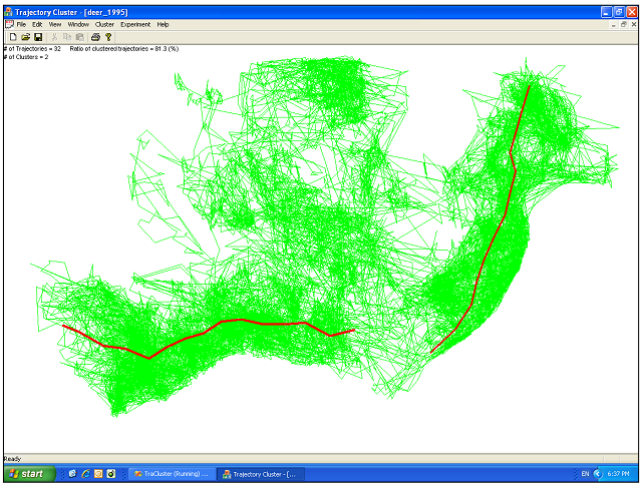
Clustering Result: *Elk1993*



ε = 27 and *MinLns* = 9 → # of clusters = 13

40

Clustering Result: *Deer1995*

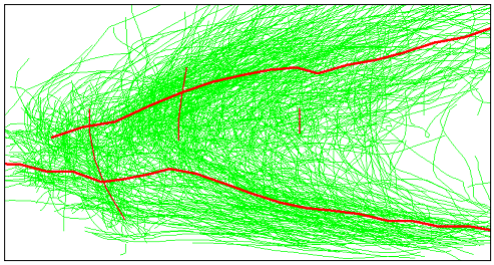


ε = 29 and *MinLns* = 8 → # of clusters = 2

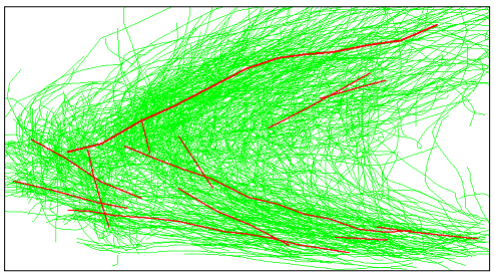
41

Effects of the Parameter Values

A larger ε or a smaller *MinLns* → a smaller number of larger clusters
e.g.,
ε = 33 and *MinLns* = 6 →
5 clusters (132 line segments)

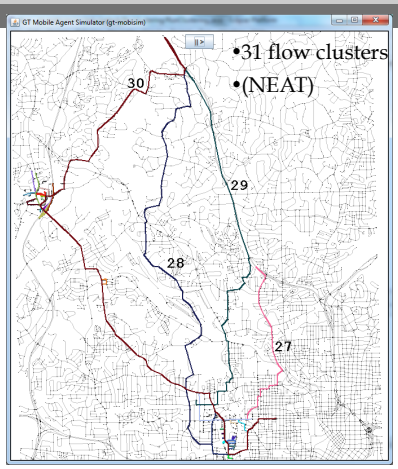


A smaller ε or a larger *MinLns* → a larger number of smaller clusters
e.g.,
ε = 26 and *MinLns* = 6 →
13 clusters (31 line segments)

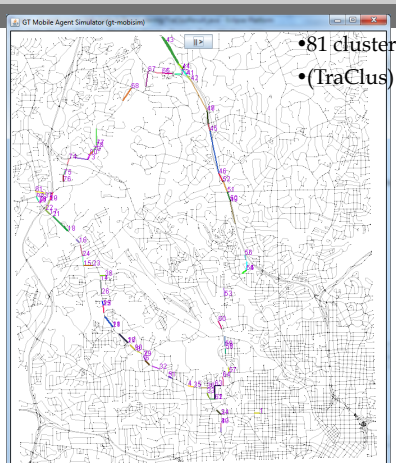


42

Problems with existing approaches



•31 flow clusters
•(NEAT)



•81 clusters
•(TraClus)

•Trajectories in road networks tend to have many sharp turns due to
•spatial constraints. Without road-network awareness, TraClus gen erates too many small clusters, high error ratio

43

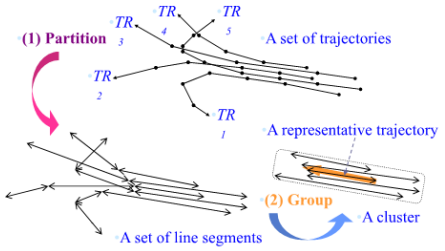
Trajectory Mining Research - State of the Art

❖ Clustering subtrajectories based on spatial correlations

➢Main idea: identify similar subtrajectories /spatial and temporal segments with hi gh density, which represent interesting movement patte rns/ areas of interest

➢TRACLUS (Lee et. al., 200 7): extends DBSCAN to clu ster sub-trajectories (partial clustering)

Two phases: ***partitioning*** and ***grouping***

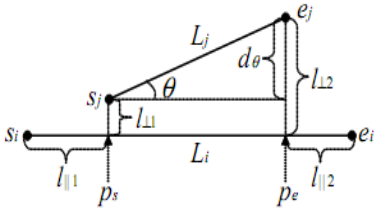


44

Measuring Similarity of Trajectories (Existing approaches)

- Criteria
  - Geometric shape / direction changes for partitioning
    - Geometric shape/ spatial proximity for grouping
- Euclidean-based distance

$$D(\tau_1,\tau_2)|_T = \frac{\int_T d(\tau_1(t),\tau_2(t))dt}{|T|},$$



$$d_\perp = \frac{l_{\perp 1}^2 + l_{\perp 2}^2}{l_{\perp 1} + l_{\perp 2}}$$

$$d_\parallel = \mathrm{MIN}(l_{\parallel 1}, l_{\parallel 2})$$

$$d_\theta = \|L_j\| \times \sin(\theta)$$

- *perpendicular distance,*
- *parallel distance*
- *angle distance*

Figure 5: Three components of the distance function for line segments.

45

Georgia Tech College of Computing
Defining the New Face of Computing

DISL cercs

NEAT: Road-network aware trajectory clustering

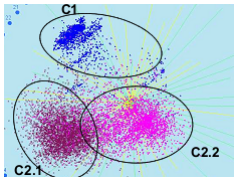
Ling Liu
College of Computing,
Georgia Institute of Technology

•Binh Han, Ling Liu, Edward Omiecinski, IEEE TMC 2014, ICDCS 2012

46

Clustering Mobile Object (MO) Trajectories

- **Goal:** Discover trajectory clusters representing major traffic flows and dense movement areas in road networks
- **Applications:** Urban planning, Public transit planning, Location-dependent advertising and entertainment
- **Unique Challenges**:
  - Distance for trajectories is different from distances for multi-dimensional points
    - Conventional data clustering algorithms fail to work with trajectories effectively
    - ➢ Harder to summarize MO trajectory data into interesting patterns through visualization



•Visualizing datasets of
•n-dimensional data points

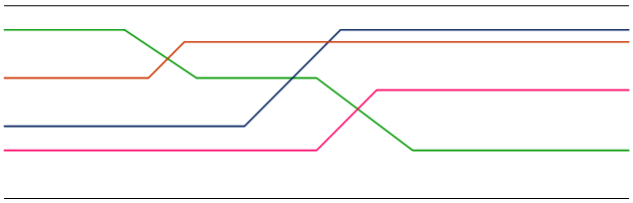
NEAT approach

- a road-NEtwork Aware Trajectory clustering (NEAT) that considers:
  - Spatial constraints in a road network
    - Road segments
    - Road intersections
    - Speed limits
  - Traffic flow of objects moving among road segments
  - Network distance based proximity, i.e. shortest path distance
- Extensive experiments show that NEAT is efficient and offers high quality clusters compared to the traditional density-based approaches.

49

2. Overview – Design guidelines

- Example 1(traffic semantics): Given a set of objects moving along the same road segment.
  - They share similar movements wrt. the road network



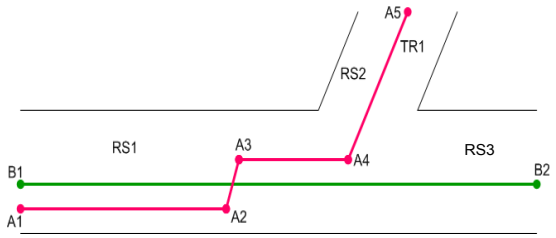
•Trajectories within the same road segment should be grouped together regardless of their shapes of movement

50

Overview – Design guidelines

- Example 2 (shape and direction changes):
  TR1 ={ A1A2, A2A3, A3A4, A4A5},
  TR2 = {B1B6}: No significant turning points

•TR2 is still needed to be split into 2 fragments at the road j unction.

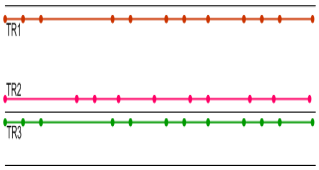


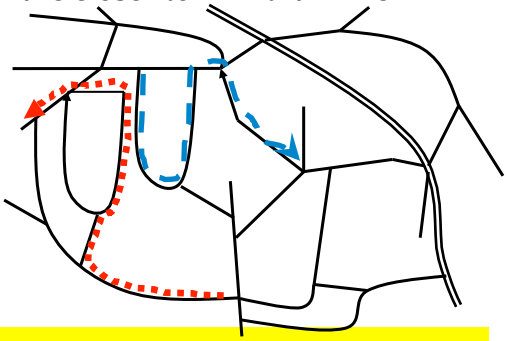
•Only at the junctions, moving objects may make dramatic change of their movement

51

Overview – Design guidelines

- Example 3 (road network proximity):
  - In the Euclidean space: TR3 is closer to TR2 than TR1
  - In a road network: TR1 are closer to TR2 than TR3





•Use shortest path distance, instead of the Euclidean distance when measuring spatial similarity

52

Overview: NEAT - Design guidelines

- Three road network aware mechanisms to capture MO trajectory similarities:
  - **Trajectory fragment:**
    - The road intersections are the initial partitioning points where a trajectory can be split into sub-trajectories, called trajectory fragments.
  - **Base Cluster:**
    - The trajectory fragments corresponding to a road segment can be viewed as a locally dense cluster of movement involved in the given set of trajectories.
  - **Flow Cluster**:
    - Merge trajectory fragments based on their continuity with regard to the traffic flows on consecutive road segments.

53

Overview – Road-network model

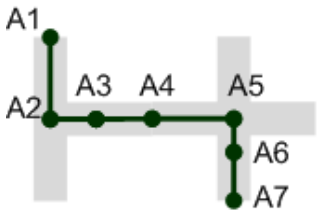
- A **road network** is modeled as a directed graph G=(V, E):
  - V ={road intersections v} (i.e., nodes)
  - E = {road segments e} (i.e., edges)
- An **edge** *e = (sid, p, q): sid:* road segment Id, *p, q:* road junctions
  - $L_p(e)$*:* set of adjacent edges of e which connect to e at p
- A road network location l is a tuple (sid, x, y):
  - sid: id of the road segment contains this point belongs
  - (x,y): geometric coordinate of the point
- A trajectory TR is a sequence of points *TR = {trid,* $l_0 l_1 ... l_n$*}* capturing the locations of a MO in a road network over time and uniquely identified by *trid*
- A sub-trajectory is a subsequence of locations in a trajectory

54

NEAT model – trajectory fragment

- t-fragment:
  - Given a trajectory $TR = \{trid, l_0 l_1 ... l_n\}$ and a road segment e, a ***t-fragment*** represents a sub-trajectory $l_k l_{k+1} ... l_{k+l}$ consisting of i+1 consecutive points in TR which lie on the same road segment, denoted by $tf = (trid, sid, l_k l_{k+i})$
  - Example:

TR = $A_1A_2 \ldots A_7$ has 3 t-fragments A_1A_2, A_2A_5 and A_5A_7



55

NEAT model – Base Cluster

- Base cluster: A base cluster is a group of t-fragments (extracted from a given set of trajectories)which are located on the same road segmen

- The **density** of a base cluster S: d(S) = # t-fragments in S (cardinality of S).
- **dense-core**(B): the base cluster in B with the highest density
- **The netflow** between two base clusters Si and Sj = the number of trajectories participating in both clusters:

$$f(S_i, S_j) = |PTr(S_i) \oplus PTr(S_j)|$$

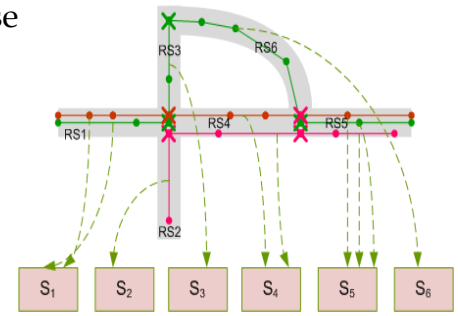
56

3.1 Base Cluster Formation

Algorithm Base_clusters_formation

- ✓Input: A set of MO trajectories T = {TR_1, TR_2, …,TR_N}
- ✓Output: A set of base

1. Trajectory splitting:
   - Trajectories → t-fragments
2. T-fragment grouping:
   - T-fragments → base clusters



57

3.2 Flow cluster formation

- Flow-based approach: merging the base clusters using density and flow based metrics to form flow clusters

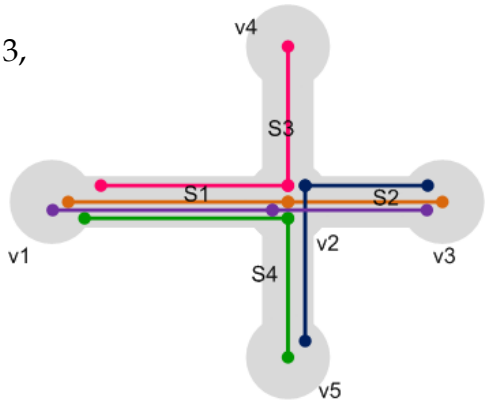
Algorithm Flow_based_clustering

- ✓Input: A set of base clusters B = {S_1, S_2, …,S_M}
- ✓Output: A set of flow clusters W = {F_1, F_2, …, F_Q}
- ✓Three tasks:
  - **Density based initialization,**
  - **f-Neighbor based merging,**
  - **Flow-correlation based termination**

58

NEAT model: An example

- A set of base clusters: B={S_1, S_2, S_3, S_4}
- Density: d(S_1) = 4, d(S_2)= 3, d(S_3) = 1, d(S_4) = 2
- S_1 =dense-core(B)
- netflow: f(S_1,S_2) = 2
- *f*-neighborhood: $N_f(S_1, v_2)$={ S_2, S_3, S_4}
- MaxFlow(S_1, v_2) = {S_2}
- F={S_1,S_2} is a flow cluster
- f(F,S_3) = 1, f(F,S_4) = 2



59

NEAT model – Flow Cluster

Let S_i be a base cluster and *p* denote one endpoint of e^{Si}(the road segment associated with S_i).

- The *f*-neighborhood of S_i *wrt. p*:

$$N_f(S_i, p)=\{S_j \mid e^{Sj} \text{ in } L_p(e^{Si}) \ \& \ f(S_i, S_j) > 0\}$$

- MaxFlow neighbor S_j of S_i wrt. *P*:

$$f(S_i, S_j)= maxflow(S_i,p) = max\ \{f(S_i, S_k)\} \mid S_k \text{ in } N_f(S_i, p)\}.$$

- A **flow cluster** is a spatially ordered list of base clusters, denoted by F = {$S_0S_1…S_N$} such that S_{i+1}in $N_f(S_i)$ and $e^{S0}e^{S1}...e^{SN}$ forms a route in the road network

60

Flow-based clustering

- Choose a base cluster to merge with another base cluster S:
  - The f-neighborhood S_j in $N_f(S, u)$ are the candidates to be merged with S
  - Consider the flow, density and speed limit of all f-neighborhood of S to decide which f-neighbor should be merged with S
    - Flow factor: $q = \frac{f(S, S_j)}{|PTr(S)|}$
    - Density factor: $k = \frac{d(S_j)}{d(S) + \sum_{S_i \in N_f(S, n_u)} d(S_i)}$
    - Speed factor: $v = \frac{speed(S_j)}{\sum_{S_i \in N_f(S, n_u)} speed(S_i)}$

61

Flow cluster formation

Choose a base cluster S_j in $N_f(S, p)$ to merge with S:

- Compute the merge selectivity of a base cluster S_j in $N_f(S, p)$,

$$SF = w_q \times q + w_k \times k + w_v \times v$$

•Flow factor density factor, speed factor

- Choose S_j if $SF(S_j)$ is the largest compared to all other f-neighborhood clusters.
- The weights $w_q > 0$, $w_k >= 0$, $w_v >= 0$ are determined by applications and $w_q + w_k + w_v = 1$

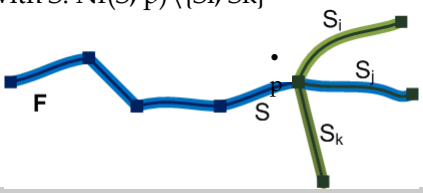
62

Flow cluster formation

- If we want to emphasize the flow factor, the weights are set as follows:

$$w_q = 1,\ w_k = w_v = 0$$

- The MaxFlow neighbor of S will be chosen to merge with S
- Each merging step: check the flow correlation:
  - Compute maxflow(S) = f(S, S_i)
  - If f(S, Si)/ f(Si, Sk) < β :
    - the flow (Si, Sk) is dominant.
    - Candidates to merge with S: Nf(S, p)\{Si, Sk}

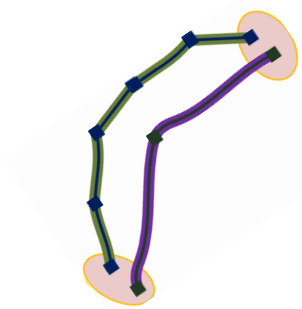


63

Flow Cluster Refinement

- Exploit opportunities to further merge some flow clusters generated from Phase 2:
  - flow clusters whose ending locations are close in terms of shortest path distance

→ Density-based grouping



64

Phase 3: NEAT Optimization → Density based Clustering Refinement

- We make an adaptation to the DBSCAN:
  (1) the data unit to be clustered is a flow cluster
  (2) no minimum cluster cardinality is required
  (3) the distance function is our modified Hausdorff distance between flow clusters:

$$\begin{aligned} dist_N(F_i, F_j) &= dist_N(r_{F_i}, r_{F_j}) \\ &= max\{max_{a\in\{a_1,a_2\}} min_{b\in\{b_1,b_2\}} d_N(a,b), \\ &\quad max_{b\in\{b_1,b_2\}} min_{a\in\{a_1,a_2\}} d_N(b,a)\} \end{aligned} \quad (5)$$

where d(a,b) is the shortest path from point a to point b

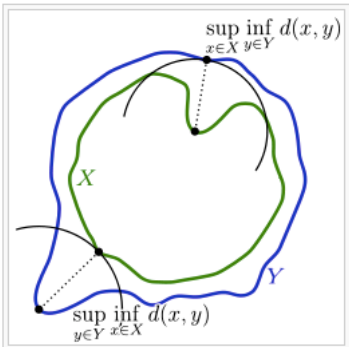
65

Hausdorff Distance

- Let X and Y be two non-empty subsets of a metric space (M, d). We define their Hausdorff distance $d_H(X, Y)$ by

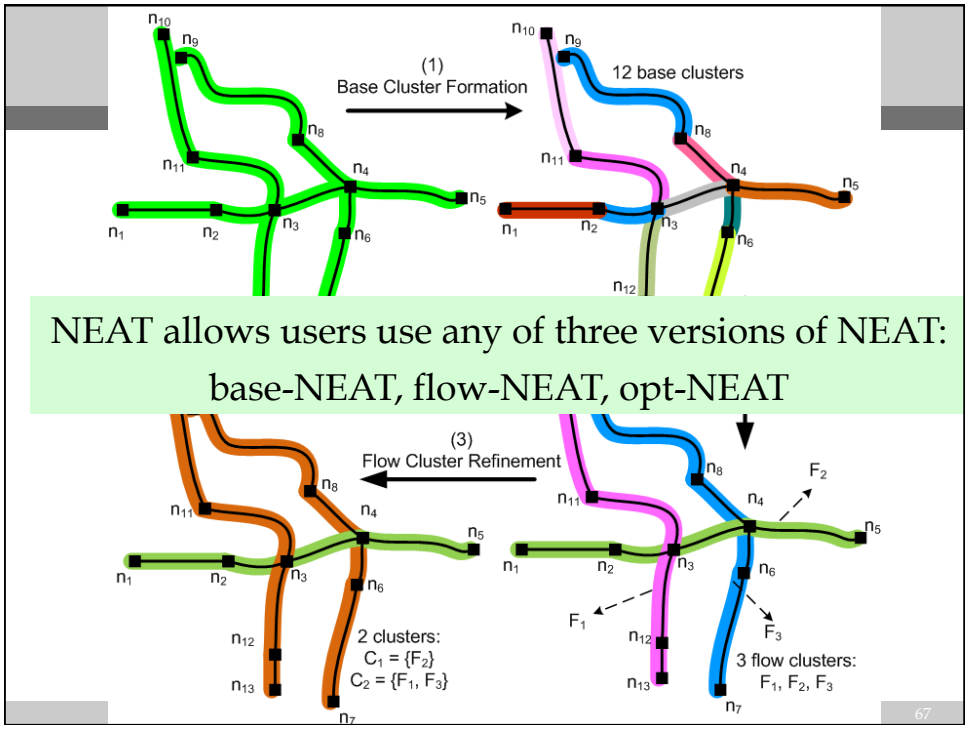
$$d_H(X,Y) = \max\{\sup_{x\in X}\inf_{y\in Y} d(x,y), \sup_{y\in Y}\inf_{x\in X} d(x,y)\},$$

where *sup* represents the supremum and *inf* the infimum.



Components of the calculation of the Hausdorff distance between the green line *X* and the blue line *Y*.

66



Computation Optimizing

- Density-based grouping:
  - For each pair of flows (F_i, F_j), compute $dist_N(F_i, F_j)$
  - $dist_N(F_i, F_j) < \varepsilon$: density-connected → grouping opportunity
- Problem: shortest path distance computation is expensive

$$O(nlogn + m)$$

- Solution: use Euclidean Lower Bound filtering to reduce the number of shortest path distance computation
  - Given the ELB: $dist_E(F_i, F_j) \leq dist_N(F_i, F_j)$
  - Only compute $dist_N(F_i, F_j)$ when $dist_E(F_i, F_j) \leq \varepsilon$
- Effective for real time trajectory clustering

68

NEAT framework: Summary

NEAT framework includes three phases:

(1) Base Cluster Formation

Each trajectory is partitioned into t-fragments.

The t-fragments which belong to the same road segment are grouped together into a *base cluster*.

(2) Flow Cluster Formation (Flow-based Clustering)

The base clusters are merged into larger clusters considering their density and continuities.

(3) Flow Cluster Refinement

We optimize the flow-based clustering results in a density-based process.

69

Experiments –Datasets and setup

- Maps: NW Atlanta, San Jose, Miami

Regions	Total length	# Segments	# Junctions	Avg. segment length	Junction degree
North West Atlanta, GA	1384.4km	9187	6979	150.7m	avg: 2.6, max: 6
West San Jose, CA	1821.2km	14600	10929	124.7m	avg: 2.7, max: 6
Miami-Dade, FL	26148.3km	154681	103377	169.0m	avg: 3.0, max: 9

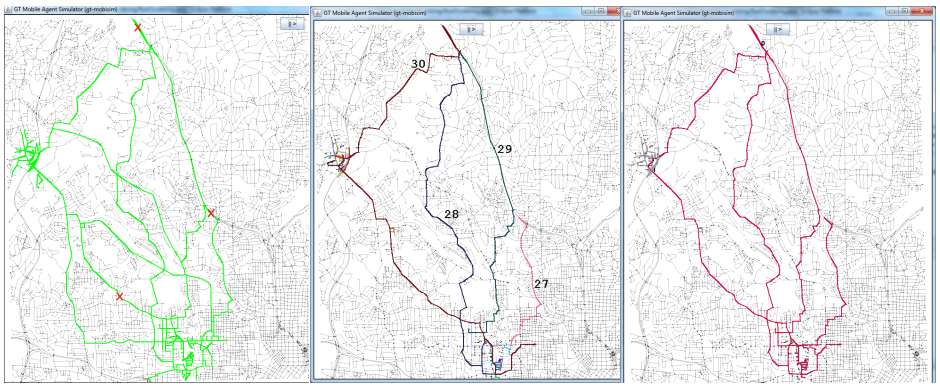
- Synthesized Datasets: generated using GT-MobiSIM trace generator

Datasets	Number of points		
	ATL	SJ	MIA
ATL/SJ/MIA500	114878	131982	276711
ATL/SJ/MIA1000	233793	255162	452224
ATL/SJ/MIA2000	468738	542598	893412
ATL/SJ/MIA3000	669924	794638	1302145
ATL/SJ/MIA5000	1277521	1296739	2262313

- Implemented using Java, run on PC Intel CPU 2.00GHz with 1 GB main memory allocated for the Java heap size

70

NEAT result visualization



•a) Input data: ATL500

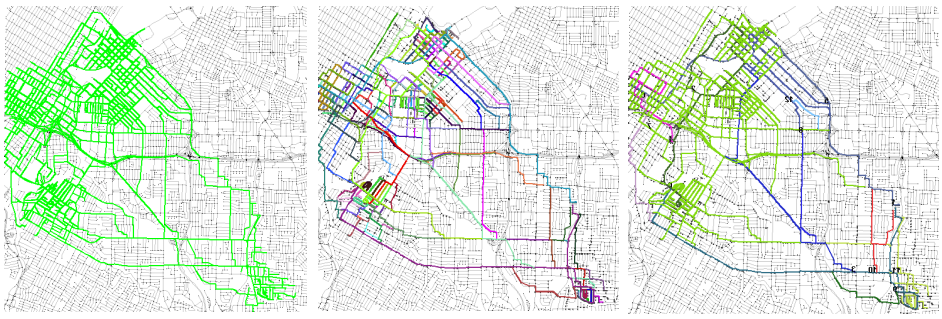
•b) 31 flow clusters

•c) 2 clusters after optimizing ph ase

•(eps=6500)

71

Results for San Jose datasets



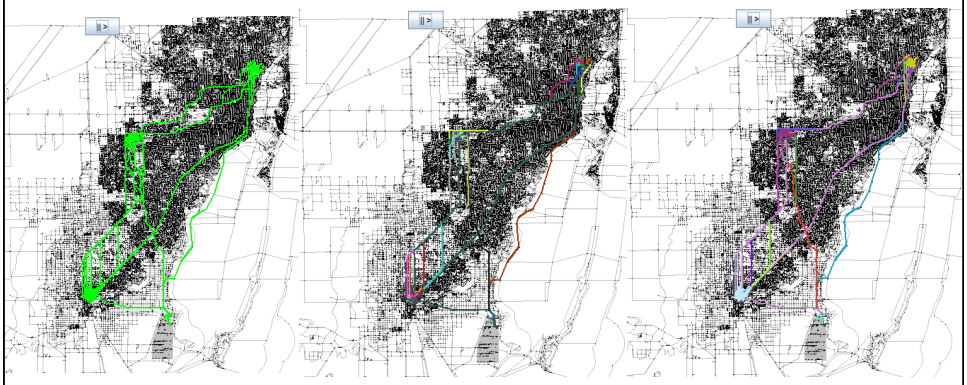
•a) Input: SJ5000

•b) 172 flow clusters

•c) 13 clusters after
•optimizing phase
•with eps = 1200

72

Results for Miami datasets



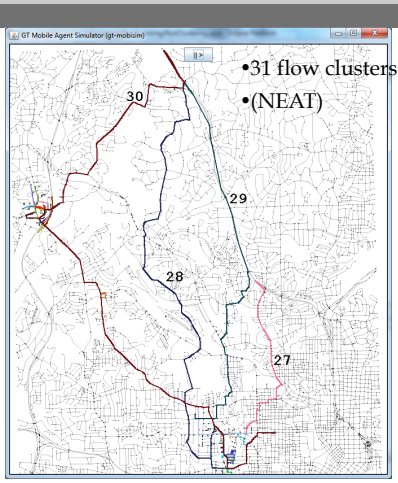
•a) Input: MIA3000

•b) 300 flow clusters

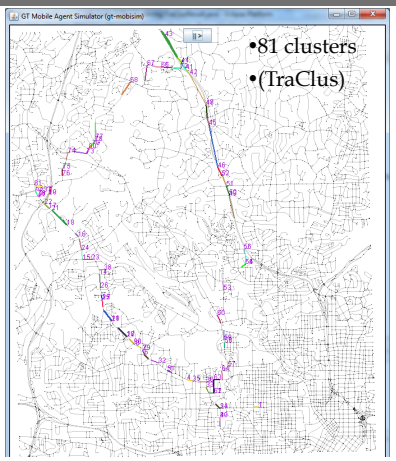
•c) 33 clusters after
•optimizing phase
•with eps = 2000

73

Efficiency and effectiveness of NEAT



•31 flow clusters
•(NEAT)

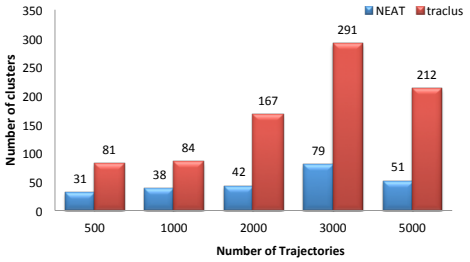


•81 clusters
•(TraClus)

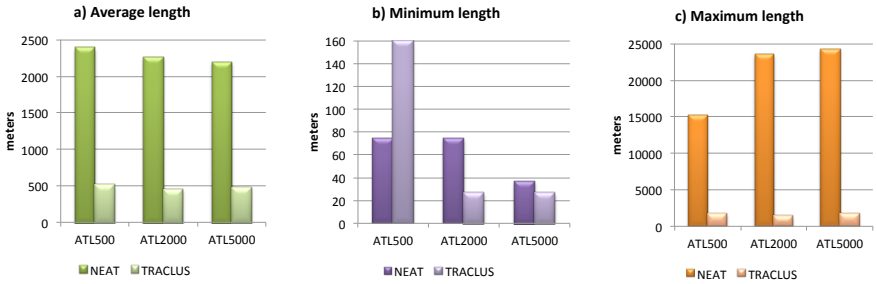
•TraClus: does not merge well, small cardinality clusters, misses important routes

74

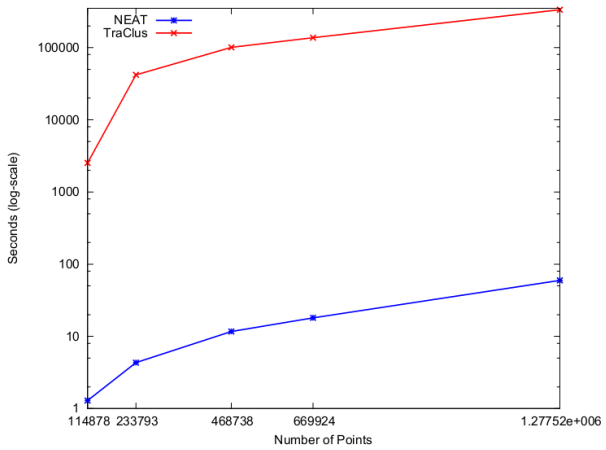
Accuracy



•NEAT merges better: trajectory clusters represent highly dense and continuous traffic flows (long routes)

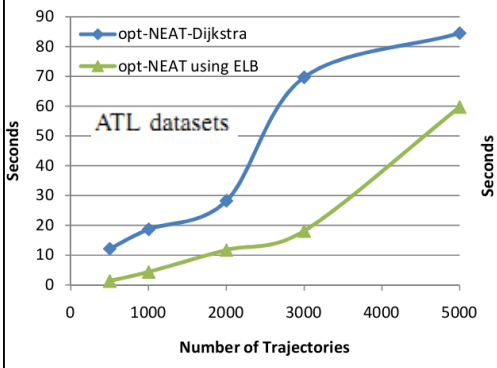


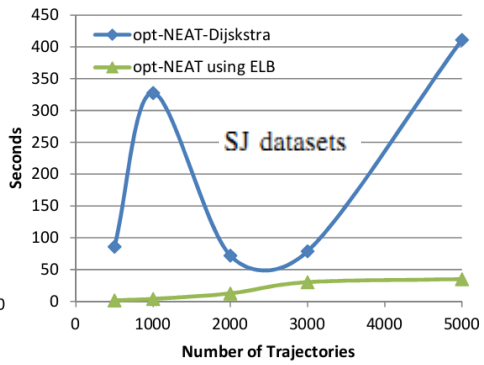
Efficiency



- NEAT is 3 orders of magnitude faster
- NEAT uses base cluster as building blocks instead of points and line segments
- No distance computation is required in basic NEAT approach

Effectiveness of using Euclidean Lower Bound

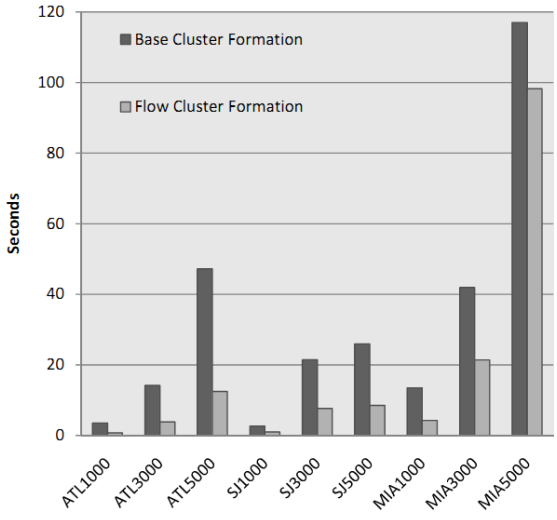




Datasets	ATL500	ATL1000	ATL2000	ATL3000	ATL5000	SJ500	SJ1000	SJ2000	SJ3000	SJ5000
# flow clusters	31	37	42	79	49	73	156	55	52	180

77

Performance of NEAT component Algorithms



78

NEAT Trajectory Clustering – Highlights

- The base cluster and flow cluster are the key elements in our NEAT model to capture the features of the traffic from the given trajectories
- *t*-fragment and *f*-neighborhood are used to identify the most critical and interesting parts of the trajectories
- NEAT devises the merging process by carefully combining the density and the continuity of traffic in the road network to improve the clustering quality

79

Thank You!

80