

# CZ4079 Final Year Project

## A Machine Learning-Based Approach to Time-Dependent Shortest Path Queries

Wei Yumou

School of Computer Science and Engineering  
Nanyang Technological University



# Agenda

- 1 Introduction
- 2 Preliminary Processing
- 3 Landmark Graph
- 4 Travel Time Estimation
- 5 Shortest Path Calculation
- 6 Conclusion and Q&A



# Agenda

- 1 Introduction
- 2 Preliminary Processing
- 3 Landmark Graph
- 4 Travel Time Estimation
- 5 Shortest Path Calculation
- 6 Conclusion and Q&A



**NANYANG  
TECHNOLOGICAL  
UNIVERSITY**  
**SINGAPORE**

# Introduction: Problem



# Introduction: Problem

- A **dynamic road network**  $G = (V, E)$  with a time-dependent weight function  $w : E, t \rightarrow \mathbb{R}$

# Introduction: Problem

- A **dynamic road network**  $G = (V, E)$  with a time-dependent weight function  $w : E, t \rightarrow \mathbb{R}$
- A **query**  $Q(u, v, t)$  that asks for a shortest path from  $u$  to  $v$  departing at time moment  $t$

# Introduction: General Approach



# Introduction: General Approach

- Traditional **Bellman-Ford or Dijkstra's algorithm** do not work with dynamic edge weights (“the curse of traditionality”)



# Introduction: General Approach

- Traditional **Bellman-Ford or Dijkstra's algorithm** do not work with dynamic edge weights (“the curse of traditionality”)
- The new **machine learning-based approach** draws on collective wisdom of thousands of taxi drivers

# Introduction: General Approach

- Traditional **Bellman-Ford or Dijkstra's algorithm** do not work with dynamic edge weights (“the curse of traditionality”)
- The new **machine learning-based approach** draws on collective wisdom of thousands of taxi drivers
- **Unsupervised learning** is employed to figure out the time-dependent edge costs

# Introduction: General Approach

- Traditional **Bellman-Ford or Dijkstra's algorithm** do not work with dynamic edge weights (“the curse of traditionality”)
- The new **machine learning-based approach** draws on collective wisdom of thousands of taxi drivers
- **Unsupervised learning** is employed to figure out the time-dependent edge costs
- A modified Dijkstra's algorithm calculates a shortest path on the fly

# Introduction: Challenges

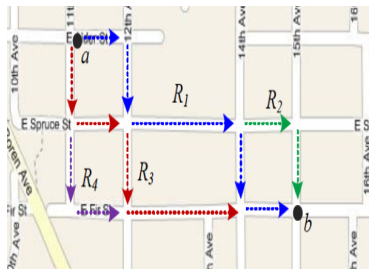


# Introduction: Challenges

- Arbitrary  $u$  and  $v$

# Introduction: Challenges

- Arbitrary  $u$  and  $v$
- Sparse sample points



# Introduction: Challenges

- Arbitrary  $u$  and  $v$
- Sparse sample points
- Limited GPS accuracy

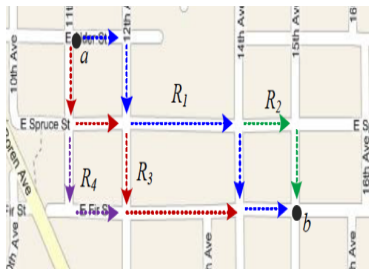


Figure 1: Examples of challenges

# Agenda

- 1 Introduction
- 2 Preliminary Processing
- 3 Landmark Graph
- 4 Travel Time Estimation
- 5 Shortest Path Calculation
- 6 Conclusion and Q&A



**NANYANG  
TECHNOLOGICAL  
UNIVERSITY**  
**SINGAPORE**



# Preliminary Processing: Data Description

- Is collected from Computational Sensing Lab at Tsinghua University
- Contains 83 million GPS records from 8,602 taxis in Beijing during May of 2009

Field	Explanation
CUID	ID for each taxi
UNIX_EPOCH	Unix timestamp
GPS_LONG	Longitude in WGS-84
GPS_LAT	Latitude in WGS-84
HEAD	Heading direction
SPEED	Instantaneous speed (m/s)
OCCUPIED	Hired (1) or not (0)

Table 1: A summary of the seven original fields

# Preliminary Processing: Reverse Geocoding

# Preliminary Processing: Reverse Geocoding

- GPS coordinate translation:  $1.34^{\circ}\text{N}$ ,  $103.68^{\circ}\text{E}$   $\rightarrow$  SCSE, NTU



# Preliminary Processing: Reverse Geocoding

- GPS coordinate translation:  $1.34^{\circ}\text{N}$ ,  $103.68^{\circ}\text{E}$   $\rightarrow$  SCSE, NTU
- China GPS shift problem: WGS84 v.s. BD09



Figure 2: An example of China GPS shift problem

# Preliminary Processing: Reverse Geocoding

- GPS coordinate translation:  $1.34^{\circ}\text{N}$ ,  $103.68^{\circ}\text{E}$   $\rightarrow$  SCSE, NTU
- China GPS shift problem: WGS84 v.s. BD09
- Solution: WGS84  $\xrightarrow{\text{Baidu API}}$  BD09  $\xrightarrow{\text{Baidu API}}$  Street



Figure 2: An example of China GPS shift problem

# Preliminary Processing: Outlier Detection

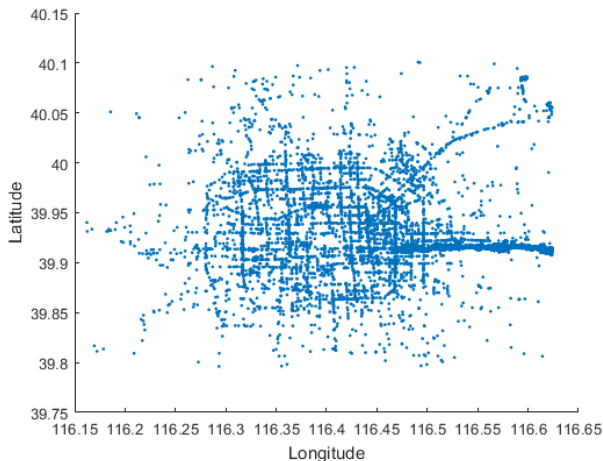


Figure 3: An example of outliers



**NANYANG  
TECHNOLOGICAL  
UNIVERSITY**  
SINGAPORE

## Theorem (*Majority Clustering Theorem*)

If a **reasonable reverse geocoder** is used to reverse-geocode a set of GPS data points which are mapped to a particular street *in reality*, then, when plotted on a 2-D plane, majority (more than 50%) of the points must be clustered together to form a rough shape that is similar to the shape of the street that they are supposed to be mapped to.

## Theorem (*Majority Clustering Theorem*)

If a **reasonable reverse geocoder** is used to reverse-geocode a set of GPS data points which are mapped to a particular street *in reality*, then, when plotted on a 2-D plane, majority (more than 50%) of the points must be clustered together to form a rough shape that is similar to the shape of the street that they are supposed to be mapped to.

Two-step procedure:

Outlier Detection = Outlier Identification + Outlier Removal



## Outlier Identification: Clustering

## **Outlier Identification:** Clustering

- Sample point concentration  $\rightarrow$  cluster concentration

## Outlier Identification: Clustering

- Sample point concentration  $\rightarrow$  cluster concentration
- Top  $k\%$  ( $k = 50$ ) largest clusters as groups of correct sample points

## Outlier Identification: Clustering

- Sample point concentration  $\rightarrow$  cluster concentration
- Top  $k\%$  ( $k = 50$ ) largest clusters as groups of correct sample points
- $10 \times 10$  self-organising feature maps implementation

## Outlier Identification: Clustering

- Sample point concentration  $\rightarrow$  cluster concentration
- Top  $k\%$  ( $k = 50$ ) largest clusters as groups of correct sample points
- $10 \times 10$  self-organising feature maps implementation

## Outlier Removal: Distance Threshold $d_{max}$

## Outlier Identification: Clustering

- Sample point concentration  $\rightarrow$  cluster concentration
- Top  $k\%$  ( $k = 50$ ) largest clusters as groups of correct sample points
- $10 \times 10$  self-organising feature maps implementation

## Outlier Removal: Distance Threshold $d_{max}$

- Assign sample points to legal centroids no farther than  $d_{max}$

## Outlier Identification: Clustering

- Sample point concentration  $\rightarrow$  cluster concentration
- Top  $k\%$  ( $k = 50$ ) largest clusters as groups of correct sample points
- $10 \times 10$  self-organising feature maps implementation

## Outlier Removal: Distance Threshold $d_{max}$

- Assign sample points to legal centroids no farther than  $d_{max}$
- Remove all “orphan” sample points

## Outlier Identification: Clustering

- Sample point concentration  $\rightarrow$  cluster concentration
- Top  $k\%$  ( $k = 50$ ) largest clusters as groups of correct sample points
- $10 \times 10$  self-organising feature maps implementation

## Outlier Removal: Distance Threshold $d_{max}$

- Assign sample points to legal centroids no farther than  $d_{max}$
- Remove all “orphan” sample points
- Use real physical distance on the Earth



## Outlier Identification: Clustering

- Sample point concentration  $\rightarrow$  cluster concentration
- Top  $k\%$  ( $k = 50$ ) largest clusters as groups of correct sample points
- $10 \times 10$  self-organising feature maps implementation

## Outlier Removal: Distance Threshold $d_{max}$

- Assign sample points to legal centroids no farther than  $d_{max}$
- Remove all “orphan” sample points
- Use real physical distance on the Earth
- Set  $d_{max} = 30\text{m}$  or  $50\text{m}$

# Preliminary Processing: Outlier Detection

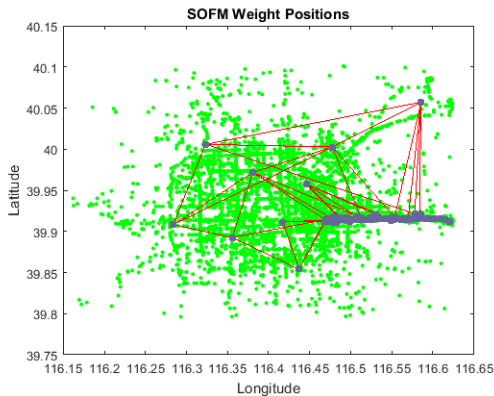


Figure 4: A plot of neuron positions after training



**NANYANG  
TECHNOLOGICAL  
UNIVERSITY**  
SINGAPORE

# Preliminary Processing: Outlier Detection

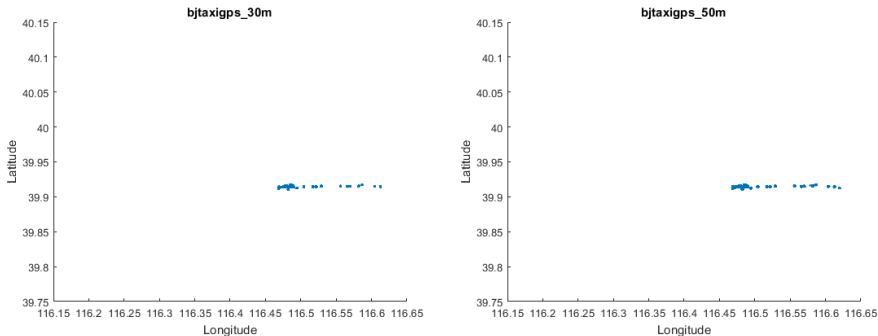


Figure 5: A plot of sample points after outlier removal

# Agenda

- 1 Introduction
- 2 Preliminary Processing
- 3 Landmark Graph
- 4 Travel Time Estimation
- 5 Shortest Path Calculation
- 6 Conclusion and Q&A



# Landmark Graph: Basic Ideas

## Definition (*Landmark*)

A landmark is a road segment that is frequently traversed by taxi drivers according to the taxi GPS trajectory database.

## Definition (*Landmark*)

A landmark is a road segment that is frequently traversed by taxi drivers according to the taxi GPS trajectory database.

## Rationale

- Sample points too sparse to model every street accurately

## Definition (*Landmark*)

A landmark is a road segment that is frequently traversed by taxi drivers according to the taxi GPS trajectory database.

## Rationale

- Sample points too sparse to model every street accurately
- Landmarks resemble common human thinking

# Landmark Graph: Basic Ideas

## Definition (*Landmark*)

A landmark is a road segment that is frequently traversed by taxi drivers according to the taxi GPS trajectory database.

### Rationale

- Sample points too sparse to model every street accurately
- Landmarks resemble common human thinking

### Step to build landmark graph

- Separate sample points into **trips**



# Landmark Graph: Basic Ideas

## Definition (*Landmark*)

A landmark is a road segment that is frequently traversed by taxi drivers according to the taxi GPS trajectory database.

### Rationale

- Sample points too sparse to model every street accurately
- Landmarks resemble common human thinking

### Step to build landmark graph

- Separate sample points into **trips**
- Count occurrences of each street

# Landmark Graph: Basic Ideas

## Definition (*Landmark*)

A landmark is a road segment that is frequently traversed by taxi drivers according to the taxi GPS trajectory database.

### Rationale

- Sample points too sparse to model every street accurately
- Landmarks resemble common human thinking

### Step to build landmark graph

- Separate sample points into **trips**
- Count occurrences of each street
- Find connections between two landmarks



# Landmark Graph: Trip Identification

CUID	UTC	GPS_LONG	GPS_LAT	OCCUPIED	TRIP_ID
1	1/5/2009 0:02:00	116.39616	39.81294	0	4552265
1	1/5/2009 0:04:00	116.39575	39.82296	0	4552265
1	1/5/2009 0:07:00	116.39567	39.82774	0	4552265
1	1/5/2009 17:08:00	116.30142	39.98105	1	1
1	1/5/2009 17:10:00	116.29514	39.98419	1	1
1	1/5/2009 17:11:00	116.28959	39.98289	1	1
1	1/5/2009 17:12:00	116.28087	39.97552	1	1
1	1/5/2009 17:16:00	116.26813	39.93537	1	1
1	1/5/2009 18:11:00	116.36537	39.95019	0	4552271
1	1/5/2009 18:12:00	116.36546	39.94886	0	4552271
1	1/5/2009 18:13:00	116.35927	39.94528	0	4552271

Table 2: An example of trip identification



**NANYANG  
TECHNOLOGICAL  
UNIVERSITY**  
SINGAPORE

# Landmark Graph: Frequency Counting

<b>CUID</b>	<b>UTC</b>	<b>GPS_LONG</b>	<b>GPS_LAT</b>	<b>Street</b>	<b>TRIP_ID</b>
1	1/5/2009 0:02:00	116.39616	39.81294	A	4552265
1	1/5/2009 0:04:00	116.39575	39.82296	A	4552265
1	1/5/2009 0:07:00	116.39567	39.82774	B	4552265
1	1/5/2009 17:08:00	116.30142	39.98105	C	1
1	1/5/2009 17:10:00	116.29514	39.98419	C	1
1	1/5/2009 17:11:00	116.28959	39.98289	C	1
1	1/5/2009 17:12:00	116.28087	39.97552	A	1
1	1/5/2009 17:16:00	116.26813	39.93537	A	1
1	1/5/2009 18:11:00	116.36537	39.95019	B	4552271
1	1/5/2009 18:12:00	116.36546	39.94886	C	4552271
1	1/5/2009 18:13:00	116.35927	39.94528	C	4552271

Table 3: An illustration of frequency counting

# Landmark Graph: Construction

For each trip

- Select a landmark  $j$
- Record intermediate streets while searching for the next landmark  $k$
- Repeat the process starting from  $k$  until all streets are examined

# Agenda

- 1 Introduction
- 2 Preliminary Processing
- 3 Landmark Graph
- 4 Travel Time Estimation**
- 5 Shortest Path Calculation
- 6 Conclusion and Q&A



**NANYANG  
TECHNOLOGICAL  
UNIVERSITY**  
**SINGAPORE**

## Definition (*Significant Edge*)

A significant edge in a landmark graph  $G = (V, E)$  is an edge  $e \in E$  that has a support at least  $m$ , where  $m$  is a parameter specified in advance.

## Definition (*Significant Edge*)

A significant edge in a landmark graph  $G = (V, E)$  is an edge  $e \in E$  that has a support at least  $m$ , where  $m$  is a parameter specified in advance.

- Build a **predictive model** for travel time of each significant edge



## Definition (*Significant Edge*)

A significant edge in a landmark graph  $G = (V, E)$  is an edge  $e \in E$  that has a support at least  $m$ , where  $m$  is a parameter specified in advance.

- Build a **predictive model** for travel time of each significant edge
- Separate weekday's travel time from weekend's

## Definition (*Significant Edge*)

A significant edge in a landmark graph  $G = (V, E)$  is an edge  $e \in E$  that has a support at least  $m$ , where  $m$  is a parameter specified in advance.

- Build a **predictive model** for travel time of each significant edge
- Separate weekday's travel time from weekend's
- Evaluate results against Baidu's estimates

# Travel Time Estimation: Underlying Distribution

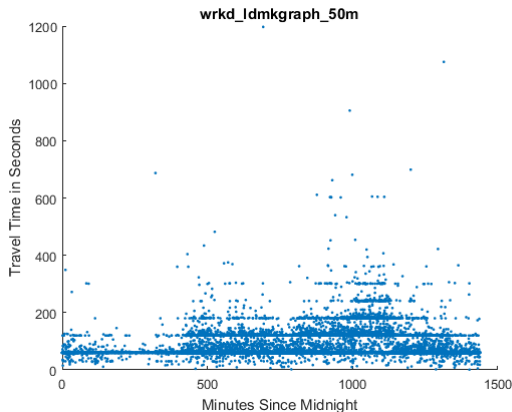


Figure 6: An example of travel time patterns

# Travel Time Estimation: Underlying Distribution

## Possible Explanations

- Drivers choose different routes to travel between the two landmarks

# Travel Time Estimation: Underlying Distribution

## Possible Explanations

- Drivers choose different routes to travel between the two landmarks
- Drivers have different driving skills, preferences and behaviours

# Travel Time Estimation: Underlying Distribution

## Possible Explanations

- Drivers choose different routes to travel between the two landmarks
- Drivers have different driving skills, preferences and behaviours
- The GPS devices report locations **periodically**, therefore, durations like 60 seconds or 120 seconds are very common

# Travel Time Estimation: Clustering

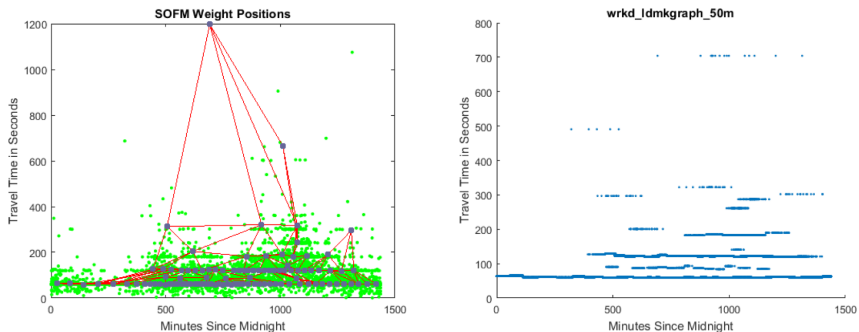


Figure 7: An illustration of travel time clustering

# Travel Time Estimation: Distribution Fit

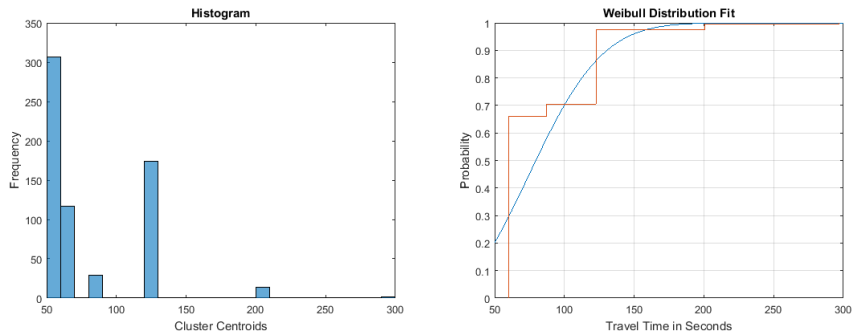


Figure 8: An illustration of fitting distribution



# Travel Time Estimation: Implementation

## Definition (*Optimism Index*)

An optimism index indicates how optimistic a driver feels about his or her driving skills. A driver with an optimism index of  $p\%$  usually drives faster than  $(1 - p)\%$  drivers under similar road conditions.

## Definition (*Optimism Index*)

An optimism index indicates how optimistic a driver feels about his or her driving skills. A driver with an optimism index of  $p\%$  usually drives faster than  $(1 - p)\%$  drivers under similar road conditions.

- Calculate and store  $\alpha$  and  $\beta$  for each 30-minute window

## Definition (*Optimism Index*)

An optimism index indicates how optimistic a driver feels about his or her driving skills. A driver with an optimism index of  $p\%$  usually drives faster than  $(1 - p)\%$  drivers under similar road conditions.

- Calculate and store  $\alpha$  and  $\beta$  for each 30-minute window
- Use optimism index  $p$  to find out travel time

# Travel Time Estimation: Evaluation

# Travel Time Estimation: Evaluation

- Select the most significant 150 edges (pairs of landmarks)

# Travel Time Estimation: Evaluation

- Select the most significant 150 edges (pairs of landmarks)
- Sample 20% of trips passing through each pair randomly

# Travel Time Estimation: Evaluation

- Select the most significant 150 edges (pairs of landmarks)
- Sample 20% of trips passing through each pair randomly
- Calculate travel time of **each edge** with optimism index  $p = 0.5$

# Travel Time Estimation: Evaluation

- Select the most significant 150 edges (pairs of landmarks)
- Sample 20% of trips passing through each pair randomly
- Calculate travel time of **each edge** with optimism index  $p = 0.5$
- Use Baidu Maps' API to give real-time estimates for **each trip**



# Travel Time Estimation: Evaluation

- Select the most significant 150 edges (pairs of landmarks)
- Sample 20% of trips passing through each pair randomly
- Calculate travel time of **each edge** with optimism index  $p = 0.5$
- Use Baidu Maps' API to give real-time estimates for **each trip**
- Compare this project's estimates with the **median** of Baidu's estimates

# Travel Time Estimation: Evaluation

- Select the most significant 150 edges (pairs of landmarks)
- Sample 20% of trips passing through each pair randomly
- Calculate travel time of **each edge** with optimism index  $p = 0.5$
- Use Baidu Maps' API to give real-time estimates for **each trip**
- Compare this project's estimates with the **median** of Baidu's estimates

Landmark Graph	RMSE	Mean Error Ratio	Mean No. of Samples Per Edge
wrkd_ldmkgraph_50m	78.84	-0.009	1824.60
wrkd_ldmkgraph_30m	87.96	-0.065	1507.56
holi_ldmkgraph_50m	87.39	-0.16	832.96
holi_ldmkgraph_30m	76.41	-0.14	681.89

Table 4: A summary of evaluation results



NANYANG  
TECHNOLOGICAL  
UNIVERSITY  
SINGAPORE

# Travel Time Estimation: Evaluation

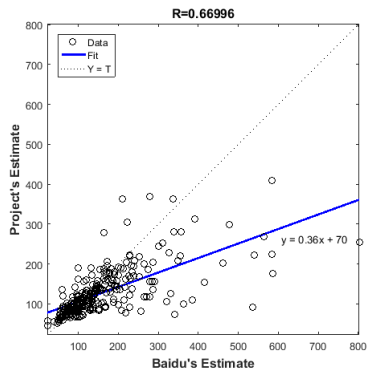
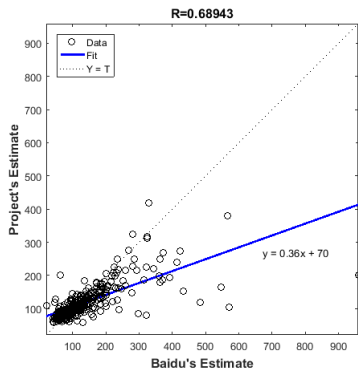


Figure 9: A plot of linear regression for weekday landmark graph

# Travel Time Estimation: Evaluation

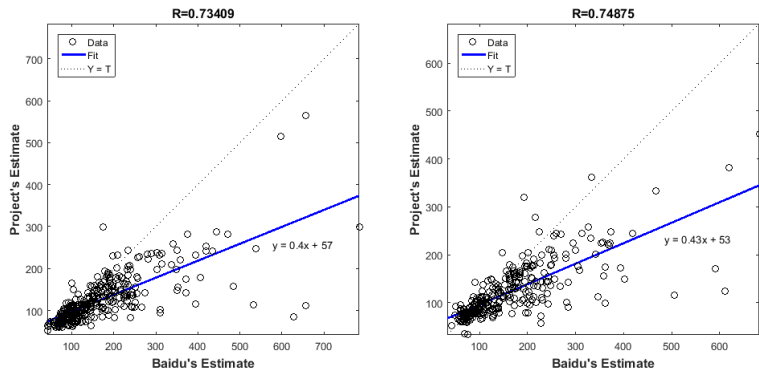


Figure 10: A plot of linear regression for weekend landmark graph

# Agenda

- 1 Introduction
- 2 Preliminary Processing
- 3 Landmark Graph
- 4 Travel Time Estimation
- 5 Shortest Path Calculation
- 6 Conclusion and Q&A



# Shortest Path Calculation: Basic Ideas

## Definition (*FIFO Graph*)

A time-dependent graph  $G = (V, E)$  with a dynamic weight function  $w : E, t \rightarrow \mathbb{R}$  is a FIFO graph iff for every edge  $(u, v) \in E$

$$\forall \Delta t \geq 0, \quad w(u, v, t_0) \leq \Delta t + w(u, v, t_0 + \Delta t) \quad (1)$$

# Shortest Path Calculation: Basic Ideas

## Definition (*FIFO Graph*)

A time-dependent graph  $G = (V, E)$  with a dynamic weight function  $w : E, t \rightarrow \mathbb{R}$  is a FIFO graph iff for every edge  $(u, v) \in E$

$$\forall \Delta t \geq 0, \quad w(u, v, t_0) \leq \Delta t + w(u, v, t_0 + \Delta t) \quad (1)$$

- Consider a landmark graph as a FIFO graph (usually true)

# Shortest Path Calculation: Basic Ideas

## Definition (*FIFO Graph*)

A time-dependent graph  $G = (V, E)$  with a dynamic weight function  $w : E, t \rightarrow \mathbb{R}$  is a FIFO graph iff for every edge  $(u, v) \in E$

$$\forall \Delta t \geq 0, \quad w(u, v, t_0) \leq \Delta t + w(u, v, t_0 + \Delta t) \quad (1)$$

- Consider a landmark graph as a FIFO graph (usually true)
- Modify Dijkstra's algorithm to calculate edge costs in an online fashion



# Shortest Path Calculation: Modified Dijkstra's Algorithm

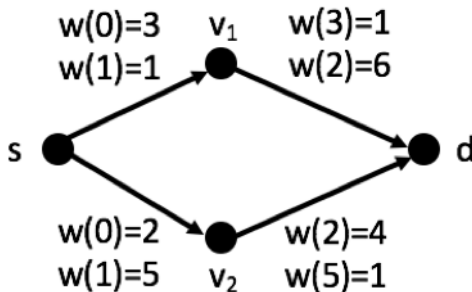


Figure 11: An example of modified Dijkstra's algorithm

# Shortest Path Calculation: Real Test Example

- Departing at 8:25 p.m.
- 19.08 mins v.s. 21 mins (Baidu's estimate)



Figure 12: An example of shortest path query

# Agenda

- 1 Introduction
- 2 Preliminary Processing
- 3 Landmark Graph
- 4 Travel Time Estimation
- 5 Shortest Path Calculation
- 6 Conclusion and Q&A



- Framed the time-dependent shortest path problem

# Conclusion and Q&A: Conclusion

- Framed the time-dependent shortest path problem
- Identified challenges in solving the problem

# Conclusion and Q&A: Conclusion

- Framed the time-dependent shortest path problem
- Identified challenges in solving the problem
- Proposed an innovative outlier detection approach



# Conclusion and Q&A: Conclusion

- Framed the time-dependent shortest path problem
- Identified challenges in solving the problem
- Proposed an innovative outlier detection approach
- Introduced algorithms for building landmark graphs

# Conclusion and Q&A: Conclusion

- Framed the time-dependent shortest path problem
- Identified challenges in solving the problem
- Proposed an innovative outlier detection approach
- Introduced algorithms for building landmark graphs
- Described procedures for estimating travel time



# Conclusion and Q&A: Conclusion

- Framed the time-dependent shortest path problem
- Identified challenges in solving the problem
- Proposed an innovative outlier detection approach
- Introduced algorithms for building landmark graphs
- Described procedures for estimating travel time
- Illustrated the modified Dijkstra's algorithm

Any questions?