# TransDec: A Big-Data Framework for Decision-Making in Transportation Systems

*Cyrus Shahabi, Ph.D.*
*Professor of Computer Science & Electrical Engineering*
*Director, Integrated Media Systems Center (IMSC)*
*Viterbi School of Engineering*
*University of Southern California*
*Los Angeles, CA 900890781*
*shahabi@usc.edu*

# OUTLINE

- Problem: Traffic Congestion
- System Solution: TransDec
- Product: ClearPath
- Research: Time-Dependent A*

# OUTLINE

- Problem: Traffic Congestion
- System Solution: TransDec
- Product: ClearPath
- Research: Time-Dependent A*

# Cost of Traffic Congestion

Traffic congestion is a **$121 billion annual drain** on the U.S. economy[1]:

- 5.5 billion lost hours
- 2.9 billion gallons of wasted fuel
- Travelers had to allow for 60 minutes to make a trip that takes 20 minutes in light traffic.

[1] Texas Transportation Institute Urban Mobility Report, 2012 data

Location data could save consumers worldwide more than $600 billion annually by 2020.
The biggest single consumer benefit will be from time and fuel savings from location-based services — tapping into real-time traffic and weather data — that help drivers avoid congestion and suggest alternative routes.

# Traffic Data Lifecycle

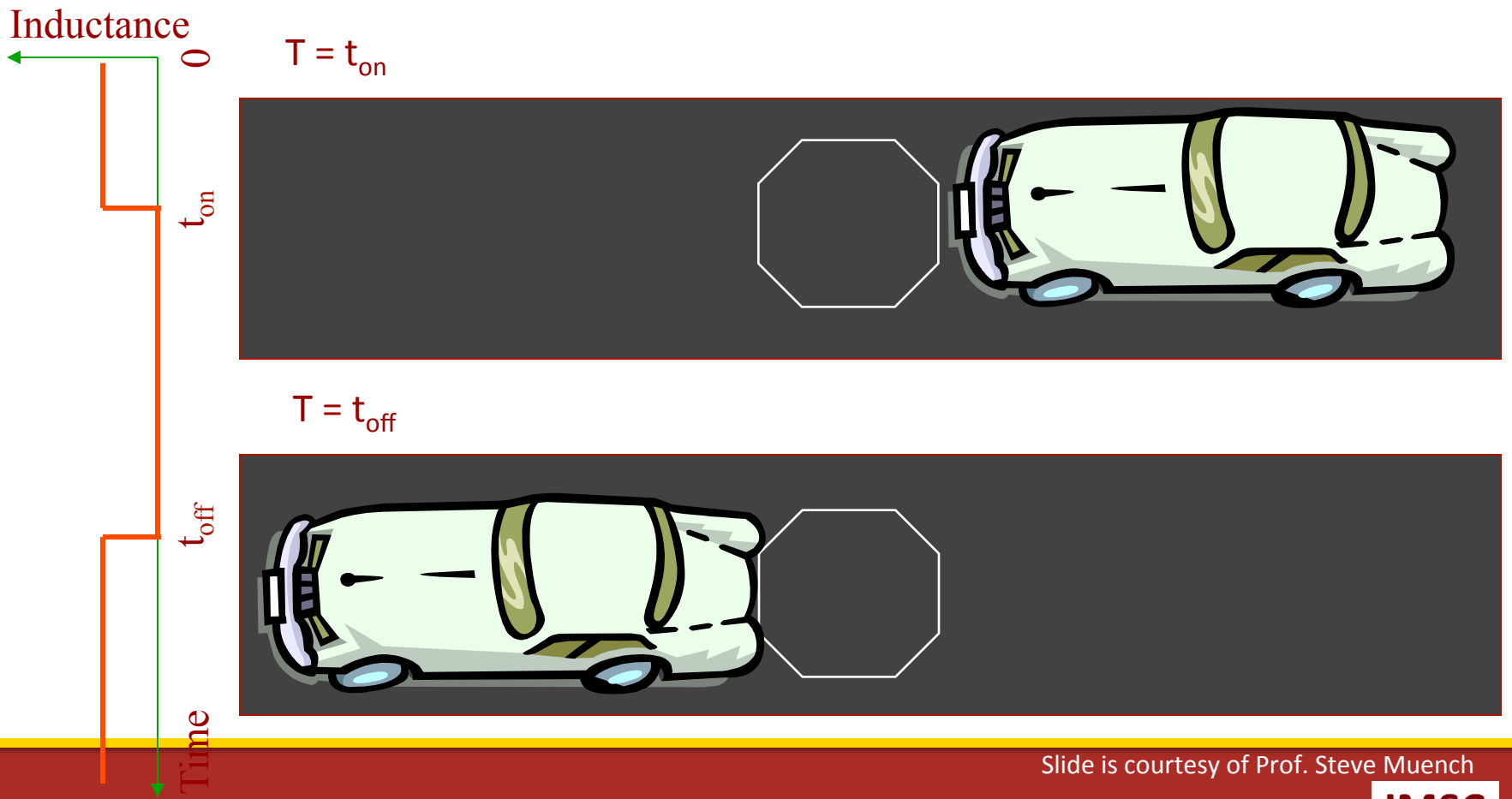
- **Loop Detectors**
  - Most commonly used traffic sensors
  - The data is collected in Detector Cabinet and relayed to the service provider
  - Provide two data fields: volume (count) and occupancy (% time a vehicle is over the sensor)

Loop Detector

Detector Cabinet

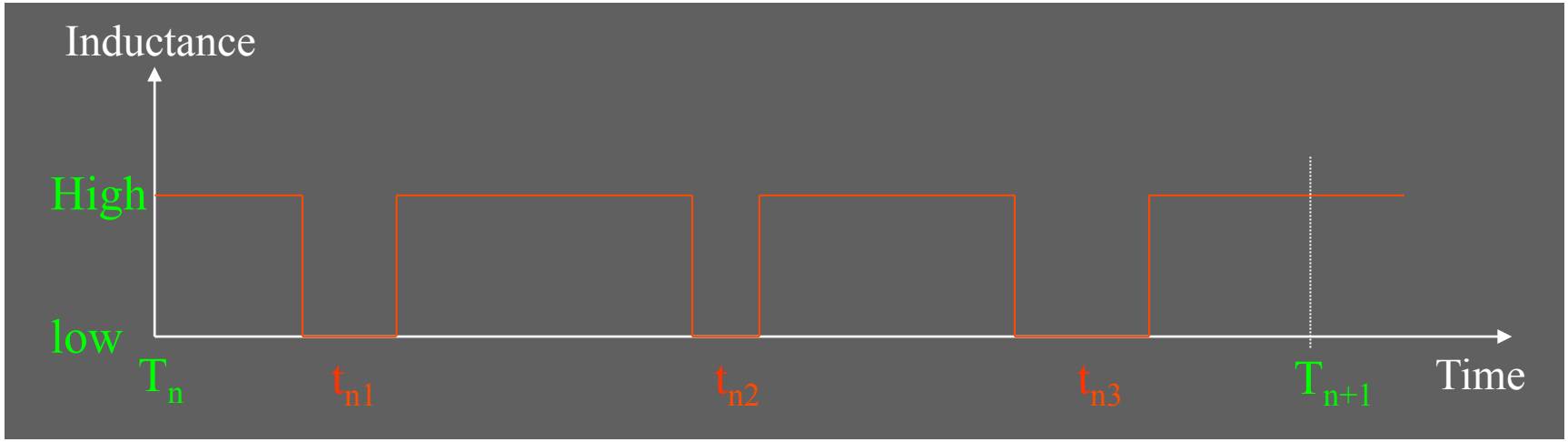
# Traffic Data Lifecycle: Loop Detectors

Loop inductance decreases when a car is on top of it.




Slide is courtesy of Prof. Steve Muench

# Traffic Data Lifecycle: Loop Detectors




- **Single loops can measure:**
  - **Occupancy (*O*): % of time loop is occupied (had a car on it) per interval**
  - **Volume (*N*): vehicles per interval**
  - **Speed = (N*L)/O where L is a constant proportional to the average length of a car**

Slide is courtesy of Prof. Steve Muench

# Traffic Data Lifecycle: Data Aggregator

## RIITS (Regional Integration of Intelligent Transportation Systems)

- A data network affiliated with Los Angeles County Metropolitan Transportation Authority (Metro)
- Collects and serves data from Caltrans, City of Los Angeles Department of Transportation (LADOT), California Highway Patrol (CHP), Long Beach Transit (LBT), Foothill Transit (FHT) and Metro

http://www.riits.net/

# An Exclusive Contract w LA-Metro

**A BIGDATA Problem: V³**

**Variety** (gps, video, loop sensor, events)

| Data Type | | | | Hourly (in KB) | Daily (in KB) | Annual (in KB) | 3 Years (in KB) |
|---|---|---|---|---|---|---|---|
| bus_mta_inv2.xml | | | | 0.96 | 23.00 | 8,395.00 | 25,185.00 |
| bus_mta_rt2.xml | 1065 | 120 | 532.50 | 31,950.00 | 766,800.00 | 279,882,000.00 | 839,646,000.00 |
| cctv_inv.xml | 57 | 86400 | 0.04 | 2.38 | 57.00 | 20,805.00 | 62,415.00 |
| cms_inv.xml | 52 | 86400 | 0.04 | 2.17 | 52.00 | 18,980.00 | 56,940.00 |
| cms_rt.xml | 48 | 75 | 38.40 | 2,304.00 | 55,296.00 | 20,183,040.00 | 60,549,120.00 |
| event_d7.xml | 11 | 75 | 8.80 | 528.00 | 12,672.00 | 4,625,280.00 | 13,875,840.00 |
| rail_mta_inv.xml | 1 | 86400 | 0.00 | 0.04 | 1.00 | 365.00 | 1,095.00 |
| rail_rt.xml | 8 | 60 | 8.00 | 480.00 | 11,520.00 | 4,204,800.00 | 12,614,400.00 |
| rms_inv.xml | 865 | 86400 | 0.60 | 36.04 | 865.00 | 315,725.00 | 947,175.00 |
| rms_rt.xml | 1236 | 75 | 988.80 | 59,328.00 | 1,423,872.00 | 519,713,280.00 | 1,559,139,840.00 |
| signal_inv.xml | 2095 | 86400 | 1.45 | 87.29 | 2,095.00 | 764,675.00 | 2,294,025.00 |
| signal_rt.xml | 2636 | 45 | 3,514.67 | 210,880.00 | 5,061,120.00 | 1,847,308,800.00 | 5,541,926,400.00 |
| tt_d7_inv.xml | 746 | 86400 | 0.52 | 31.08 | 746.00 | 272,290.00 | 816,870.00 |
| tt_d7_rt.xml | 152 | 60 | 152.00 | 9,120.00 | 218,880.00 | 79,891,200.00 | 239,673,600.00 |
| vds_art_d7_inv.xml | 115 | 86400 | 0.08 | 4.79 | 115.00 | 41,97 | |
| vds_art_d7_rt.xml | 45 | 60 | 45.00 | 2,700.00 | 64,800.00 | 23,652,00 | |
| vds_art_ladot_inv.xml | 2538 | 86400 | 1.76 | 105.75 | 2,538.00 | 926,370.00 | 2,779,110.00 |
| vds_art_ladot_rt.xml | 969 | 60 | 969.00 | 58,140.00 | 1,395,360.00 | 509,306,400.00 | 1,527,919,200.00 |
| vds_fr_d7_inv.xml | 957 | 86400 | 0.66 | 39.88 | 957.00 | 349,305.00 | 1,047,915.00 |
| vds_fr_d7_rt.xml | 361 | 30 | 722.00 | 43,320.00 | 1,039,680.00 | 379,483,200.00 | 1,138,449,600.00 |
| **Total KB from XML data** | **13980** | **864660** | **6,985.28** | **41** | | **,885.00** | **11,012,906,655.00** |

**Velocity**

**Volume**

# OUTLINE

- Problem: Traffic Congestion
- System Solution: TransDec
- Product: ClearPath
- Research: Time-Dependent A*

# ***TransDec:***

## Big data acquisition, storage & access

Input Traffic Data

Data Processing

Storage

Retrieval, Analysis &Visualization

RIITS — Los Angeles County

46 MB/min

25 TB/year

Highway (4313)

Arterial (4780)

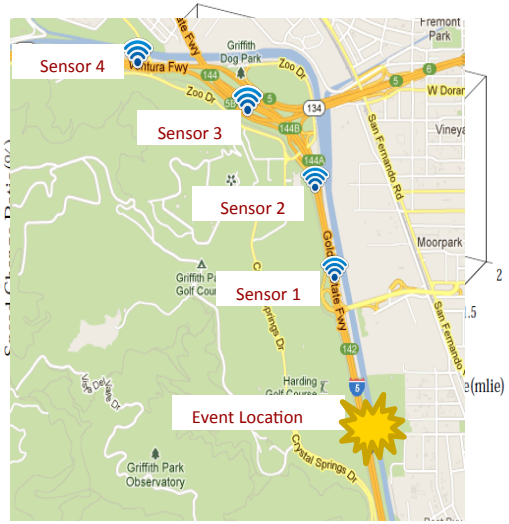
Bus & Rail (2000)

Ramp meter

Events & CMS (800/day)

**Real-time Queries & Data Cleansing**

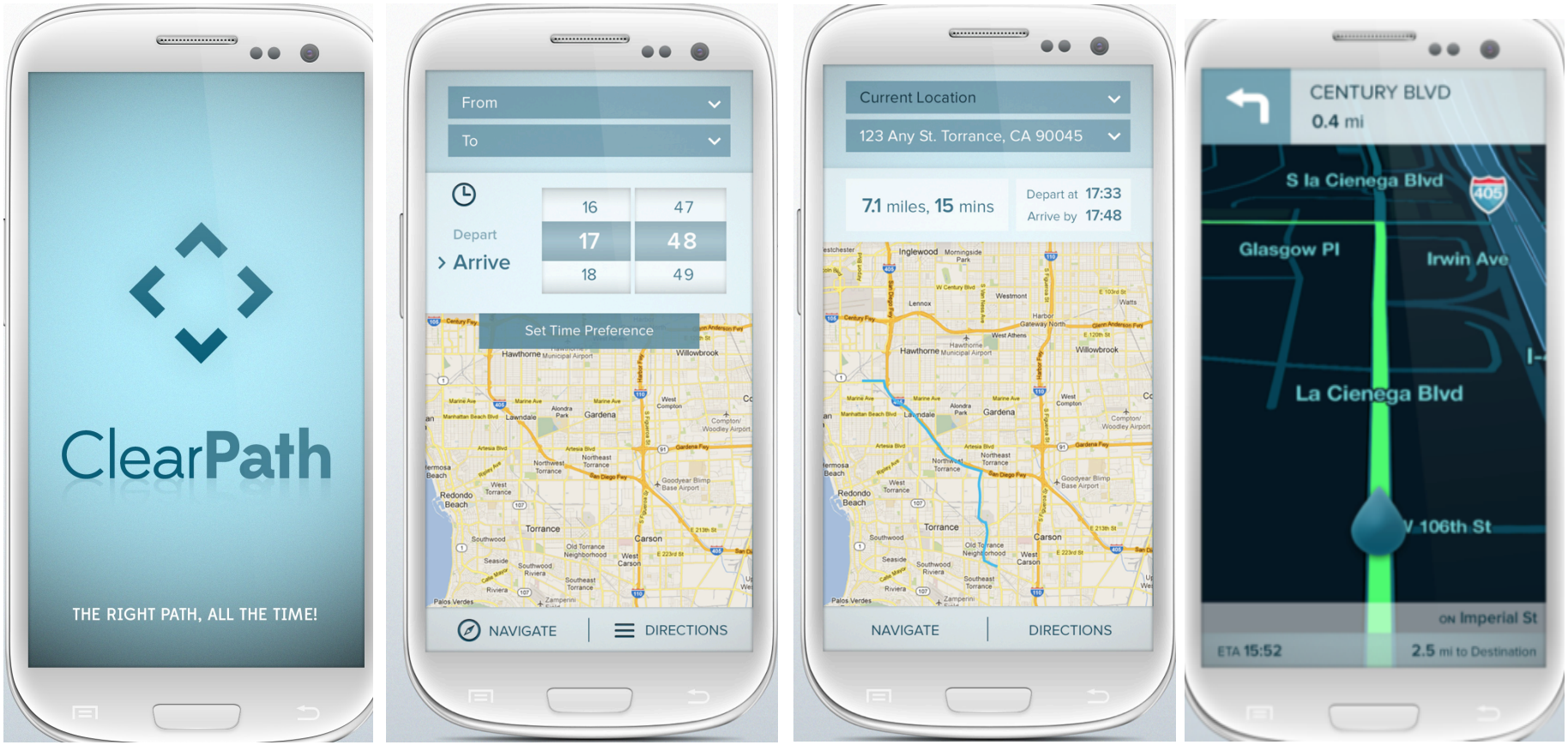
**Spatiotemporal Indexing** *(Oracle Award, IEEE CloudCom Best paper)*




**E.g., Accident impact analysis & prediction (ICDM'12)**

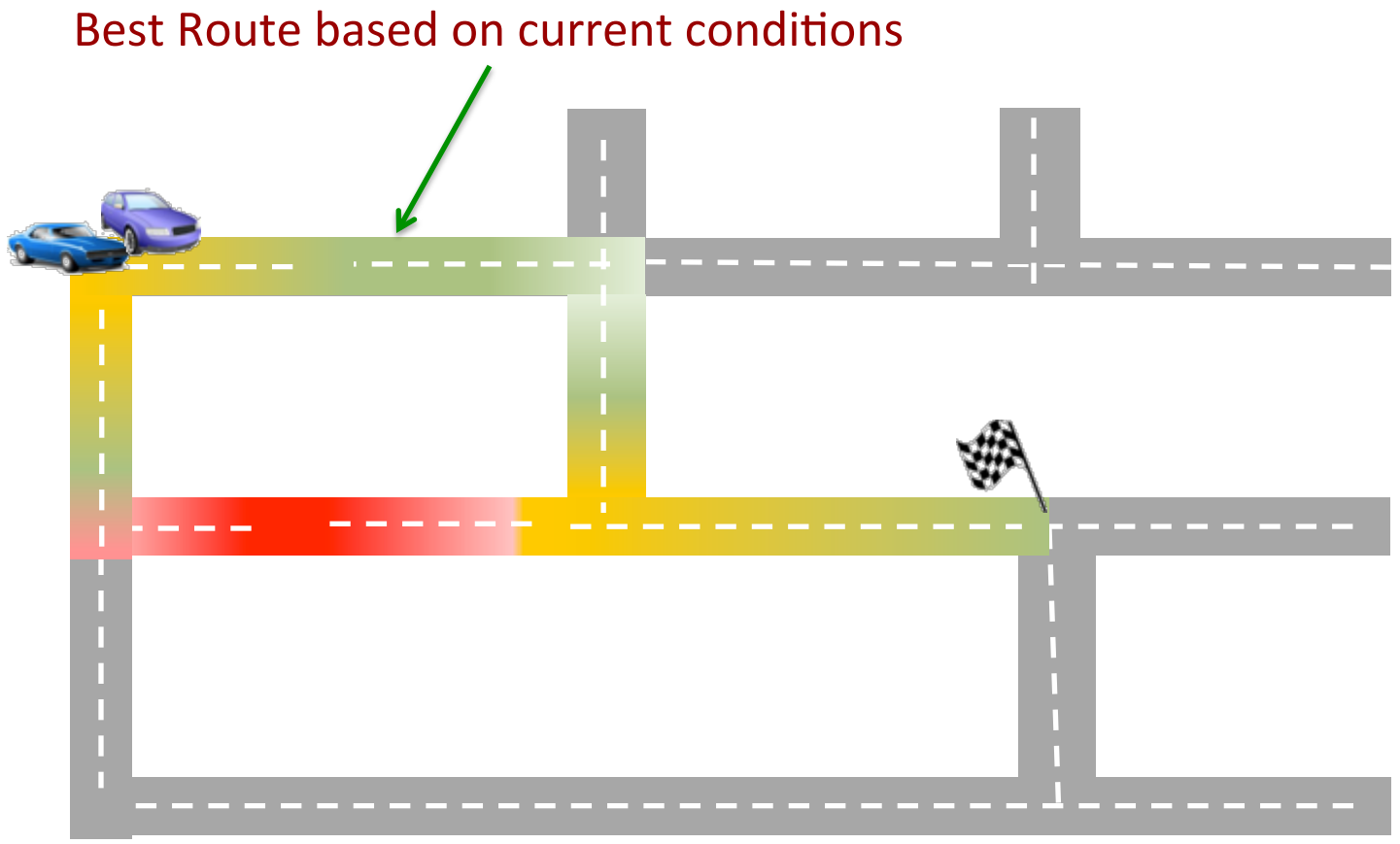
intel ORACLE Microsoft

# Product: ClearPath

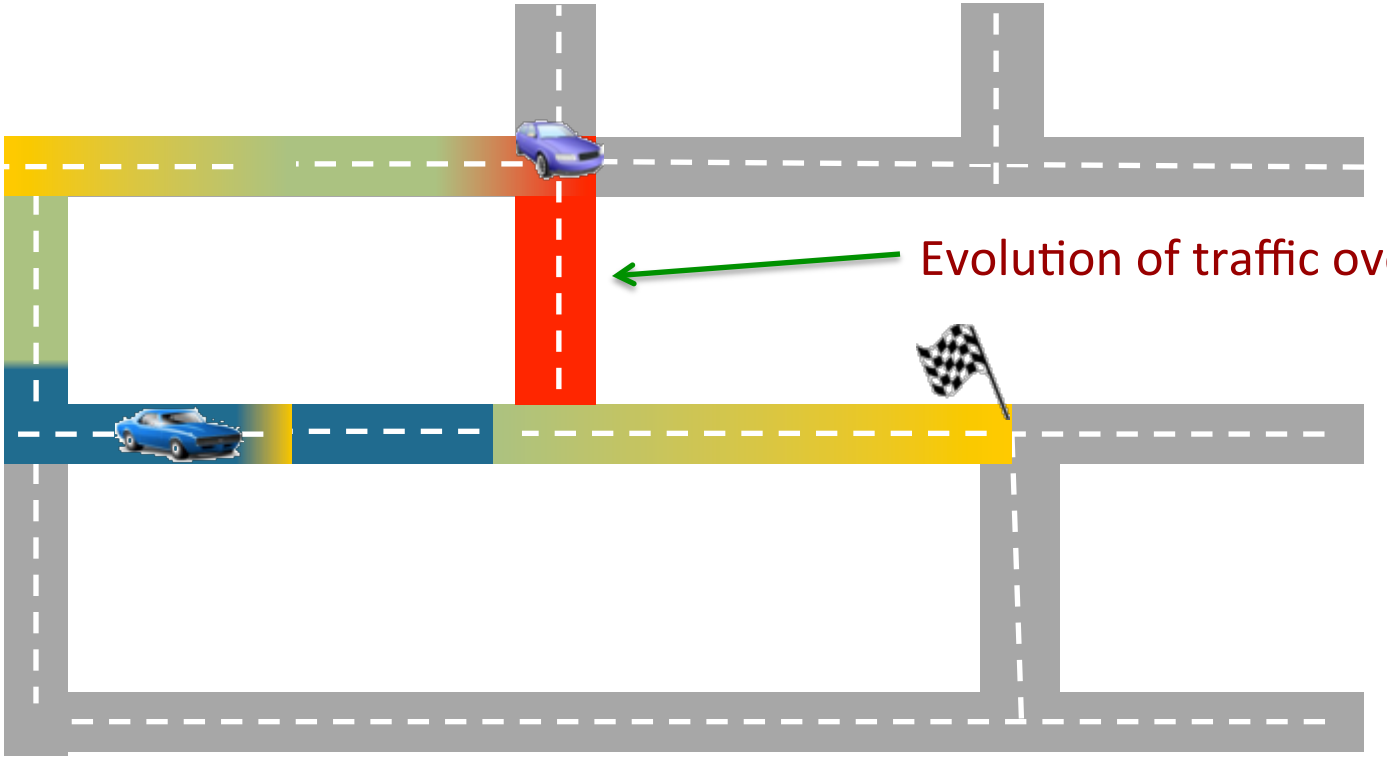



## Main Differentiator: Predictive Path Planning

# Predictive vs. Real-Time Path-Planning

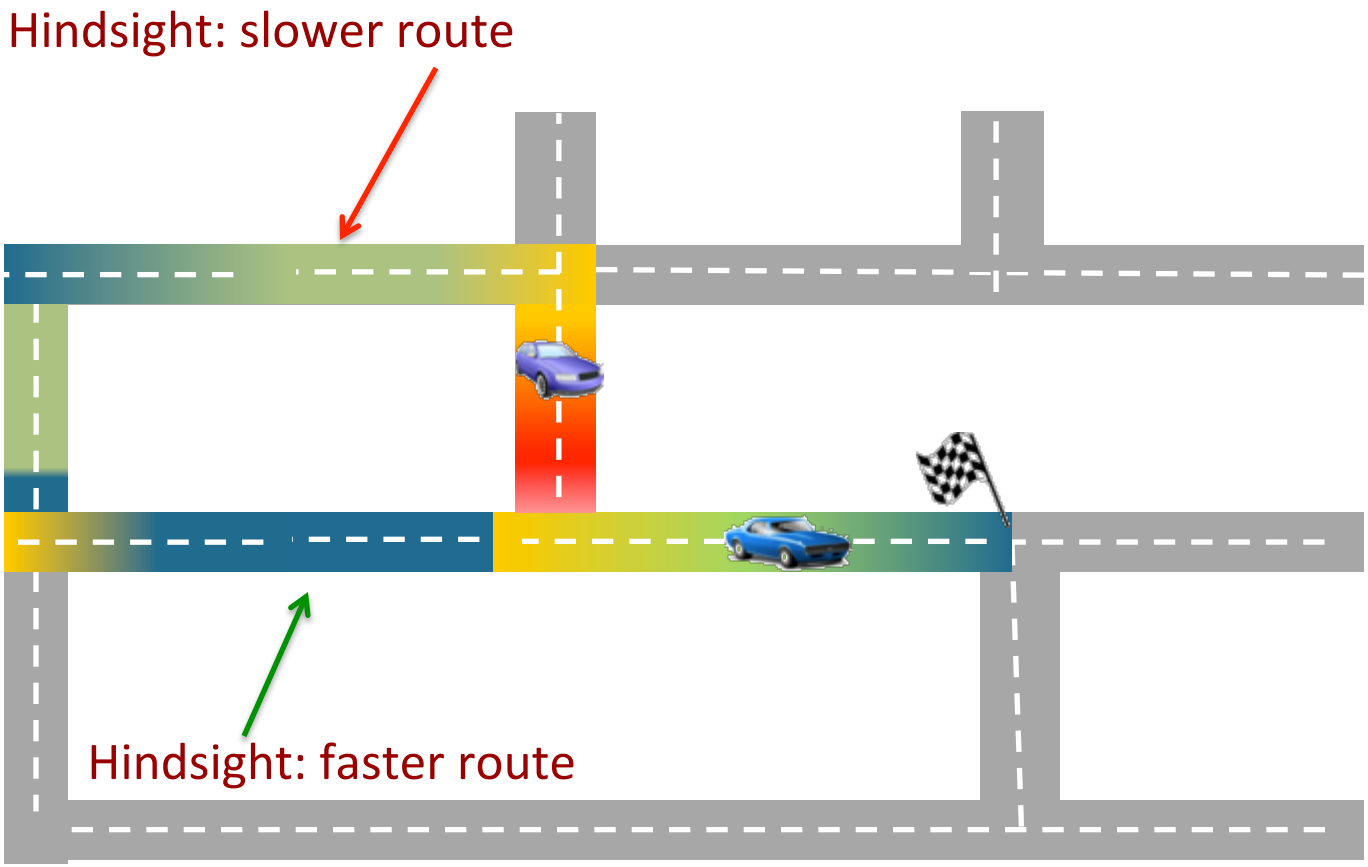


# Predictive vs. Real-Time Path-Planning




7:15AM

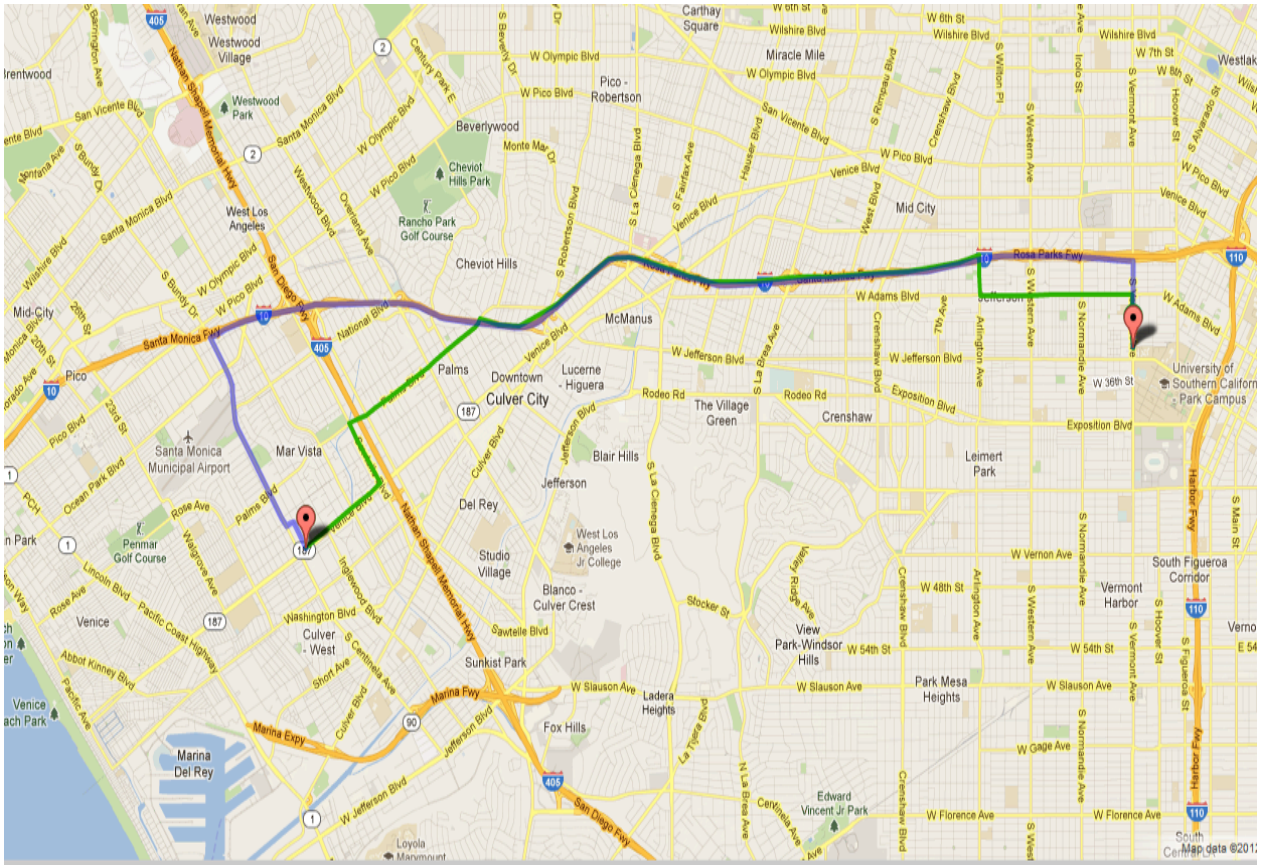
# Predictive vs. Real-Time Path-Planning




7:20AM

# Comparisons (Better Path)

## Venice → USC



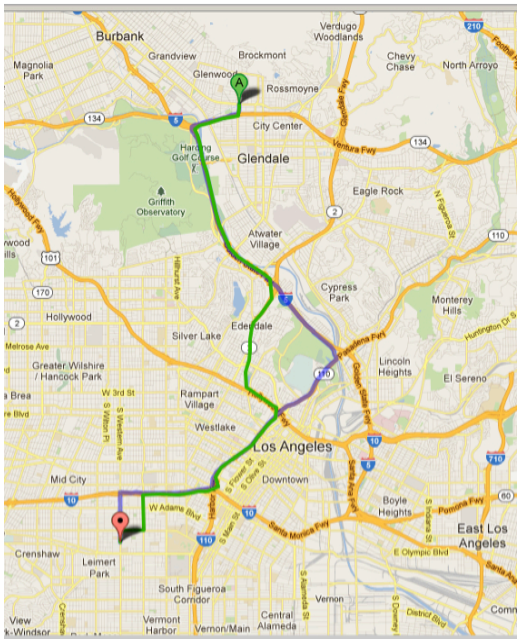

**8:30 AM**

**ClearPath:20min**

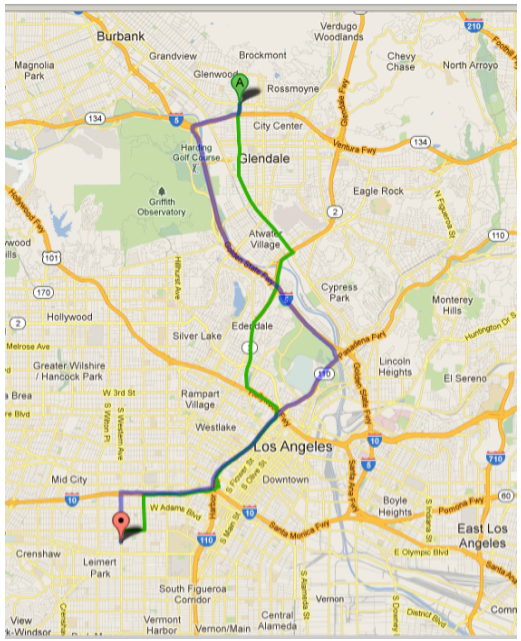
**Google: 17 min in theory, 26 min in traffic**

# Comparisons (Saved Time)

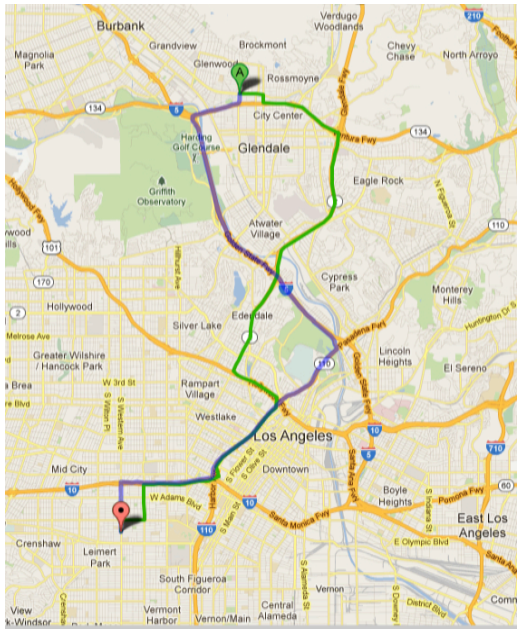
## Glendale → USC




**6:30 AM**
**ClearPath:22min**
**Google:21min, 42min w traffic**



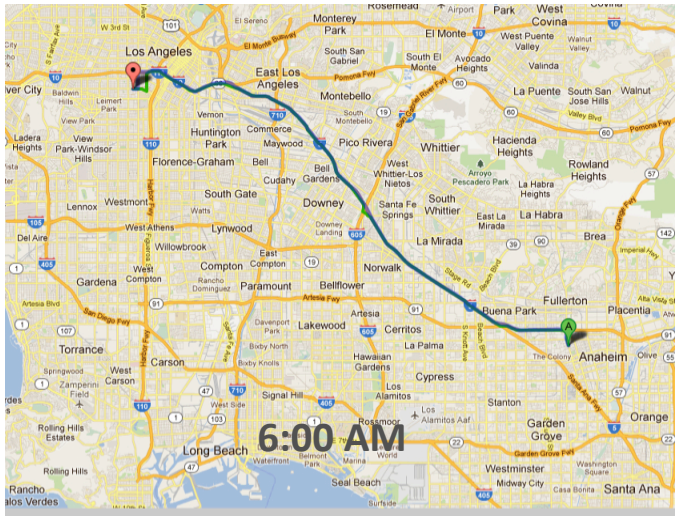

**7:15 AM**
**ClearPath:26min**
**Google:21min, 42min w traffic**

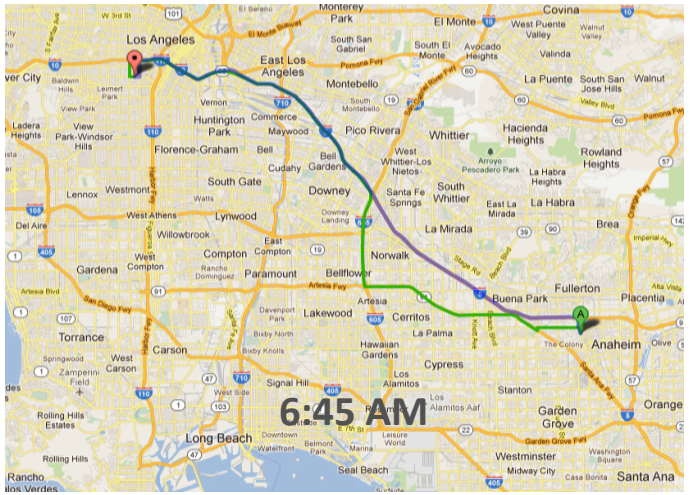


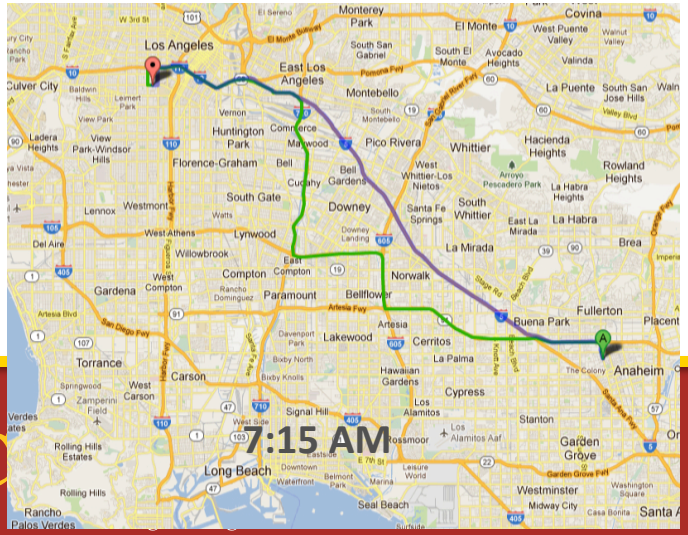

**8:30 AM**
**ClearPath:31min**
**Google:21min, 42min w traffic**

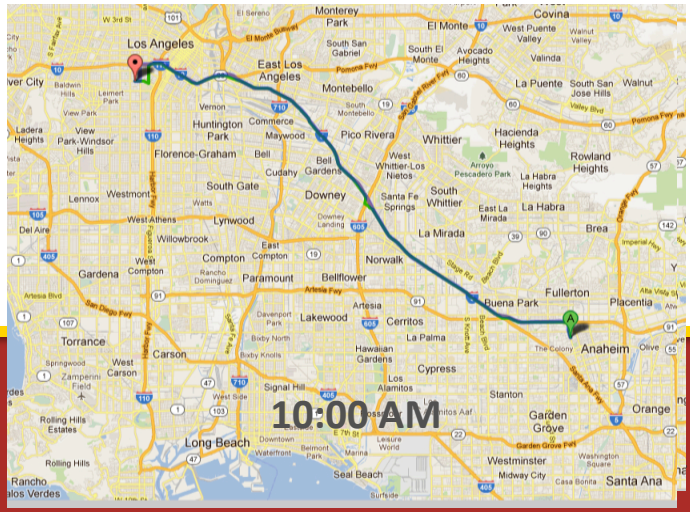
# Comparisons (Path Alternatives)

## Anaheim → USC



# Voice of America

http://www.voanews.com/content/traffic-technology-clearpath/1616682.html

# OUTLINE

- Problem: Traffic Congestion
- System Solution: TransDec
- Product: ClearPath
- Research: Time-Dependent A*

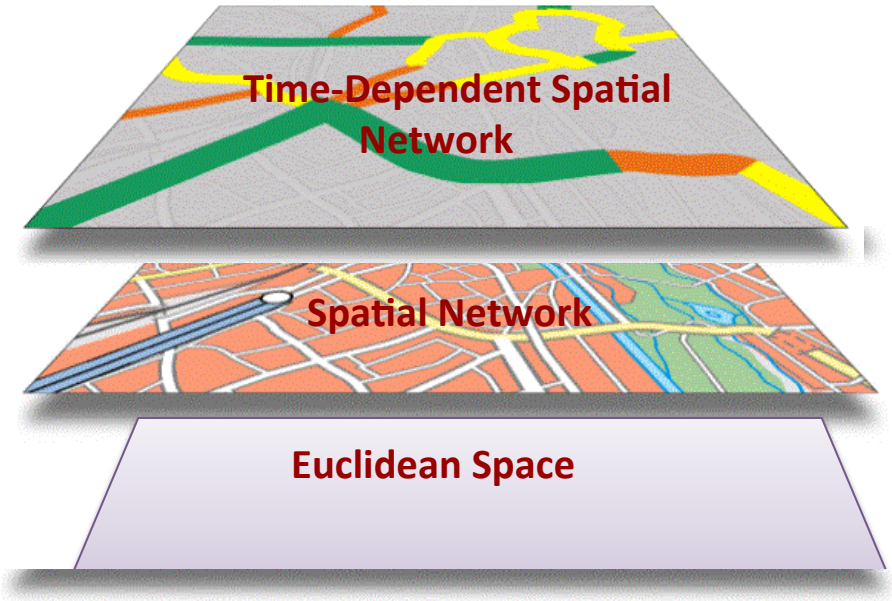
# Outline

- Distance Computation
- Motivation
- Related Work
- Time-dependent  A* Search
- Experimental Evaluation

# Outline

- Distance Computation
- Motivation
- Related Work
- Time-dependent A* Search
- Experimental Evaluation

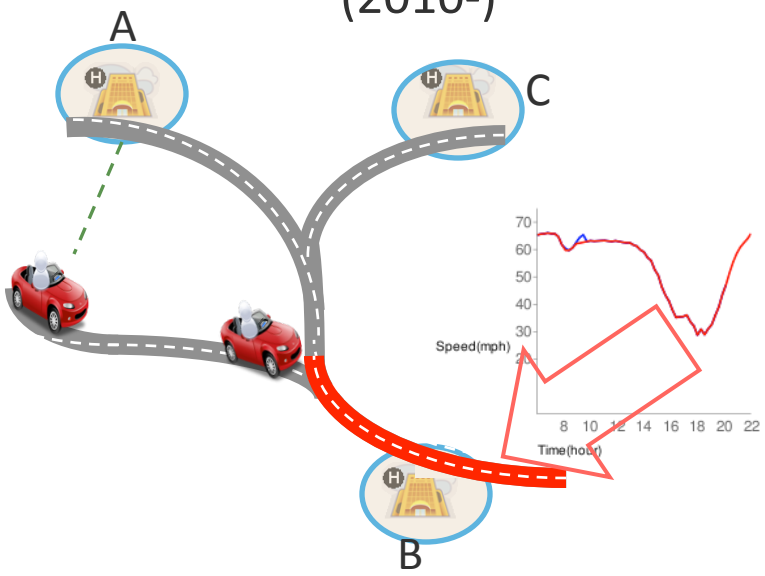
# Distance Computation




Time-Dependent Spatial Network (2010-)

Spatial Network (2003-2010)

Euclidean Space (1995-2003)




Edge weight change with time

Edge weight is constant

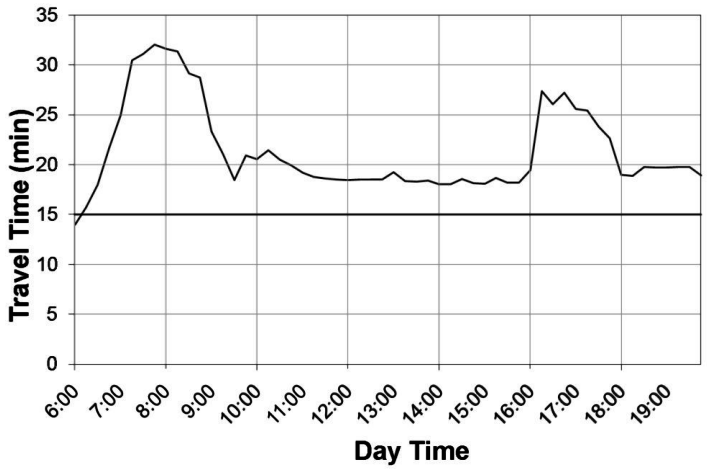
# Outline

- Distance Computation
- Motivation
- Related Work
- Time-dependent A* Search
- Experimental Evaluation

# Motivation

- Shortest-path research (2003-2010)
  - Find shortest-path based on the **constant** edge weights for each edge, (i.e., usually the maximum allowed speed-> minimum travel-time)
- In Real-world
  - The weight of an edge is a function of time, i.e., **time-dependent.**
  - Arrival-time to an edge determines the travel-time on that edge.

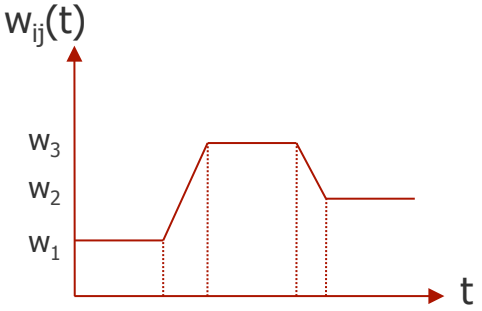

Pictures courtesy : http://www.wfrc.org/cms




Monday travel-time on a segment of I-10 in LA
(generated based on two years of historical traffic sensor data)

# Problem Definition

- Given a time-dependent spatial network where edge weights are function of time





*Source **s** and Destination **d***

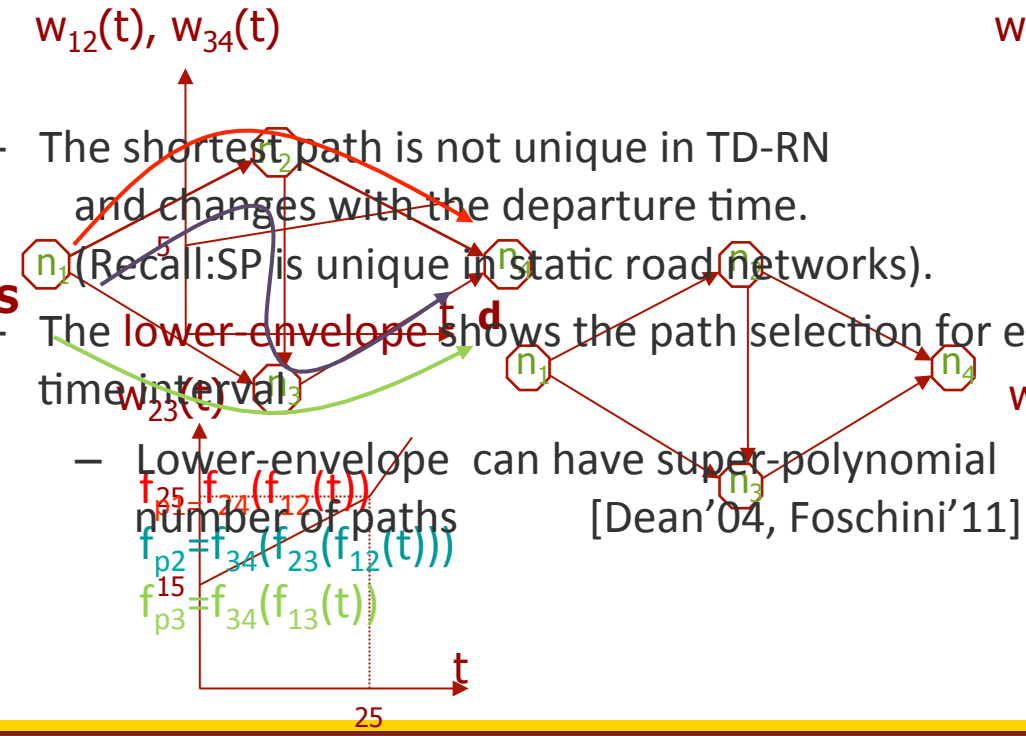
***Time-dependent Fastest Path (TDFP)***

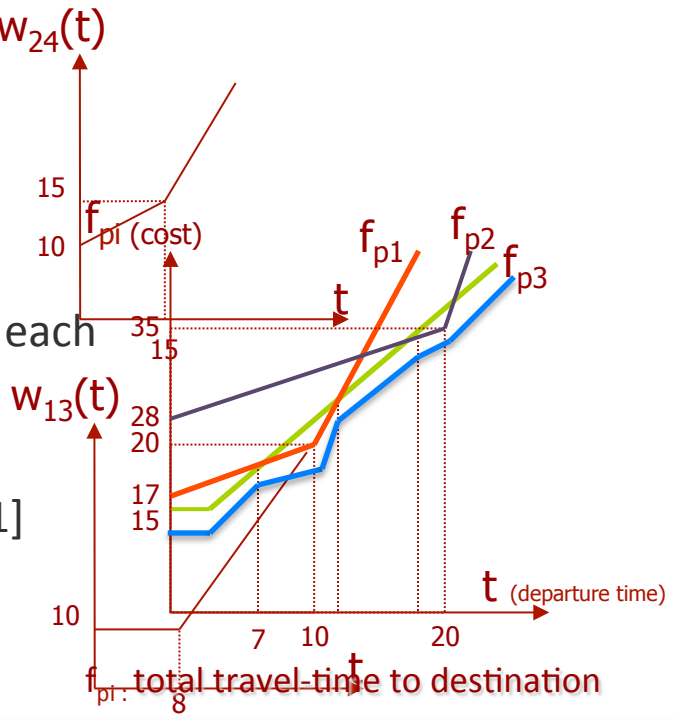
TDFP (s, d, t_s) with respect to s, d and query time t_s finds *minimum travel time path* among all paths between s and d

Challenge: Too big of a graph to find optimal path in real-time
Typical Approach: Pre-computation

# Challenges

- Is Pre-computation feasible?
  - Compute and store all distance values between all pairs of nodes
  - The shortest path is not unique in TD-RN and changes with the departure time. (Recall:SP is unique in static road networks).
  - The lower-envelope shows the path selection for each time interval
    - Lower-envelope can have super-polynomial number of paths [Dean'04, Foschini'11]

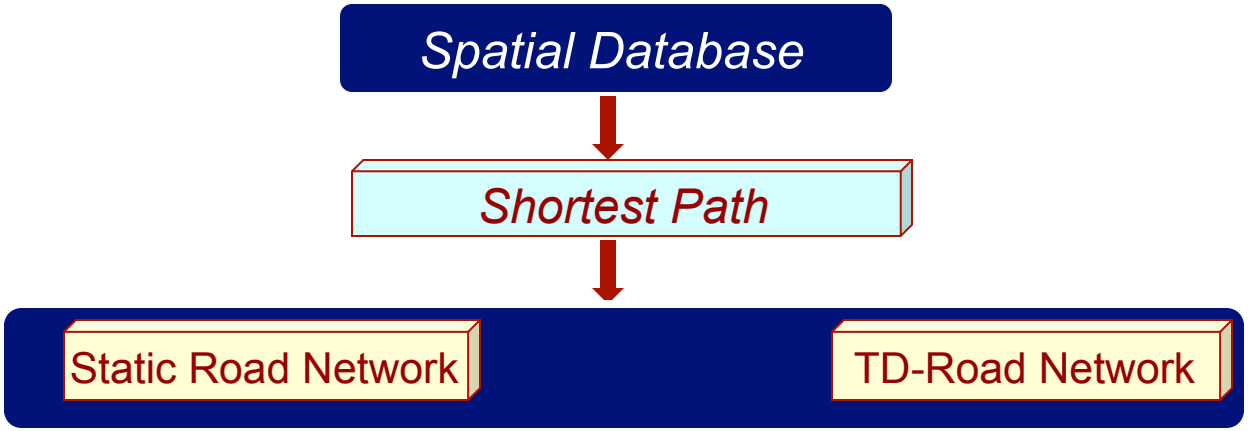


# Outline

- Distance Computation
- Motivation
- Related Work
- Time-dependent A* Search
- Experimental Evaluation

# Related Work

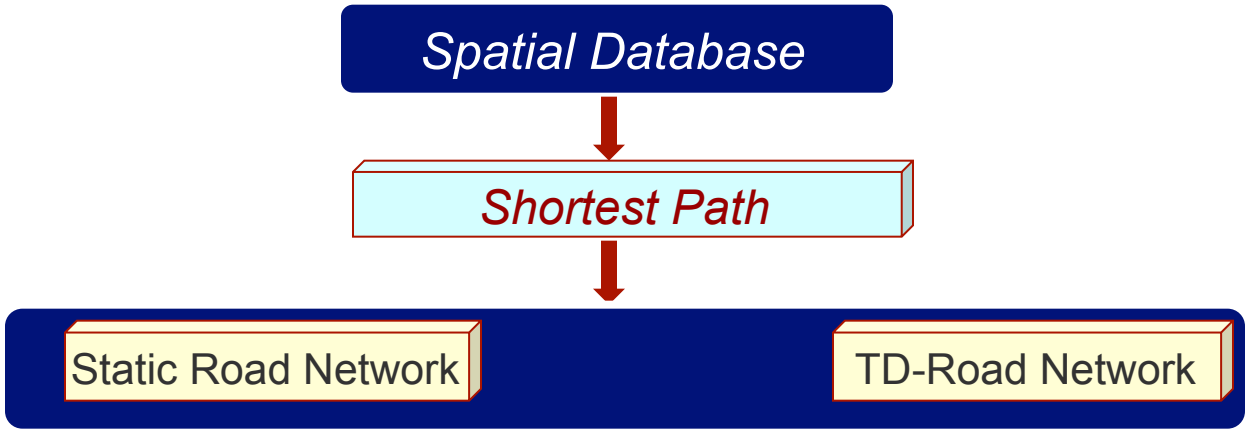



- Dijkstra [Numerische Mathematik 1959]
- A* [Hart, Nilsson & Raphael [Trans SSC 1968]

**Precomputation:**

- Geometric speed-up techniques for finding SP, [Wagner et al.,ESA'03]
- Engineering fast route planning algorithms, [Sanders et al., WEA'07]
- Hierarchical routing in RN, [Geisberger et al., WEA'08, Sanders ESA'06]
- SILC: Scalable network distance browsing [Samet et al., SIGMOD'08]
- Distance oracles for spatial networks [Sankaranarayan et al., TKDE'10]
- TEDI: Efficient Shortest Path Query Answering on Graphs [Wei, SIGMOD'11]

# Related Work




- Cooke & Halsey [JMAA'66]
- Dreyfus [OR'69] (Dijkstra Variant)
- Orda and Rom, [JACM'90] (Bellman F.)

**Precomputation:**

**Inefficient:** high storage cost and long precomputation time

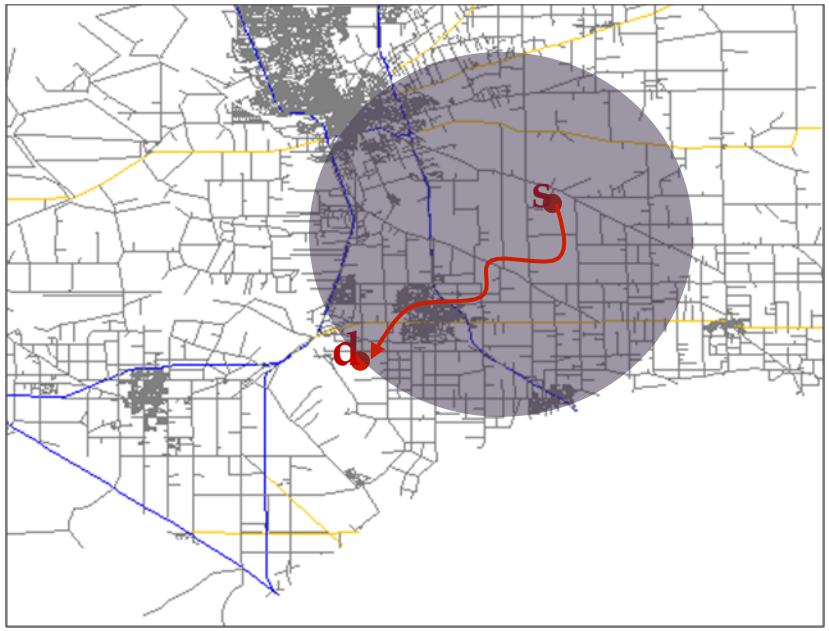
- Time-dependent SHARC [ Delling et al., ESA'09]
- Time-dependent Contraction Hierarchies [Batz et al. ALENEX'08]
- Time-dependent ALT [Delling & Wagner, WEA'07]
- Distributed Time-dependent *CH* [Kieritz et al., SEA'10]
- Core Routing on Dynamic TD RN [Delling et. al, INFORMS'11]

# Outline

- Distance Computation
- Motivation
- Related Work
- Time-dependent A* Search
- Experimental Evaluation

# Preliminaries

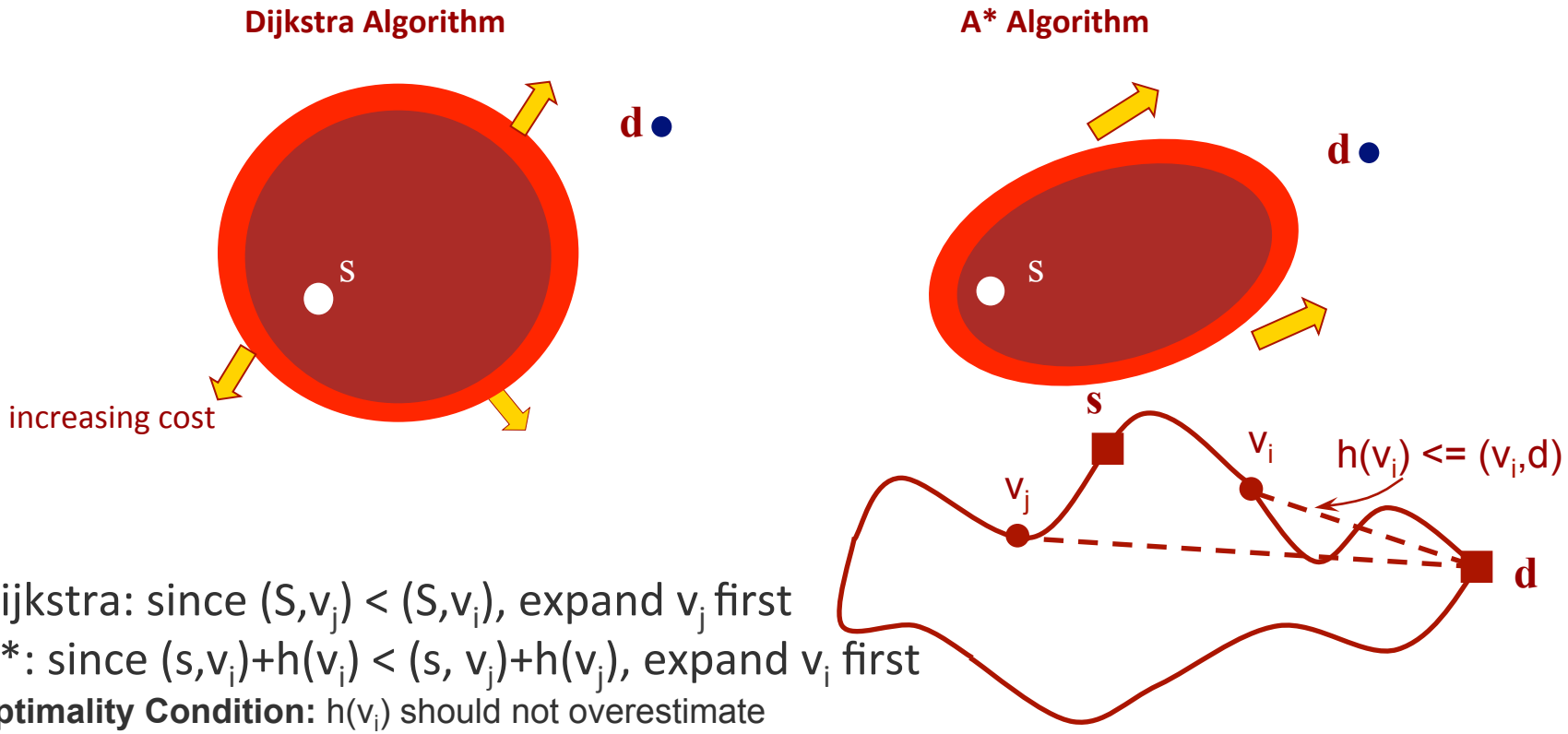
- The Dijkstra Algorithm
  - **Greedy Algorithm:** Starting from s, the network nodes reachable from s in every direction are visited in order of their distance to source




Problem: 48% of network nodes are scanned

# Preliminaries

- Dijkstra vs. A*



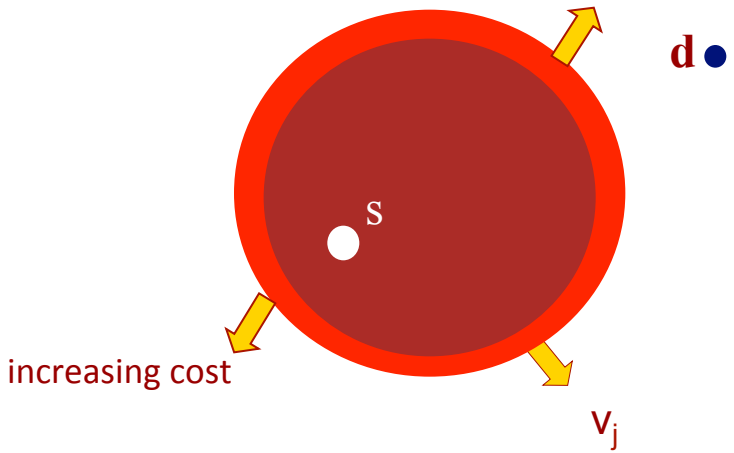

Dijkstra: since $(S,v_j) < (S,v_i)$, expand $v_j$ first

A*: since $(s,v_i)+h(v_i) < (s, v_j)+h(v_j)$, expand $v_i$ first

**Optimality Condition:** $h(v_i)$ should not overestimate the actual distance between $v_i$ and d.

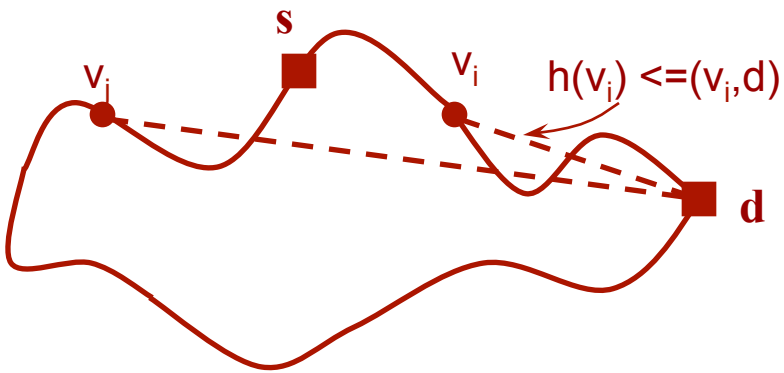
# Preliminaries

- The time-dependent shortest path problem can be solved by modifying Dijkstra Algorithm **[Dreyfus'69]**
  - **Greedy Algorithm:** Starting from s, the network nodes reachable from s in every direction are visited in order of their ***arrival-time***



# Time-dependent A* Search

- **Challenge**: Finding heuristic function $h(v_i,d) <= D(v_i,d,t)$ in TD Networks




  - The distance (travel-time) between any node $v_i$ and d changes in Time-dependent Road Networks
  - $h(v_i,d)$ also needs to be time-dependent

# Time-dependent A* Search

- **Naïve Heuristic Function:**

$$\frac{D_{EUC}(v_i, d)}{\max(speed)}$$

*Euclidean distance between $v$ and $d$ divided by the maximum speed among the edges*

  - Guaranteed to be a lower-bound as the distance between $v$ and $d$ is never overestimated
  - Problem: It is a very loose bound, hence yields insignificant performance improvement

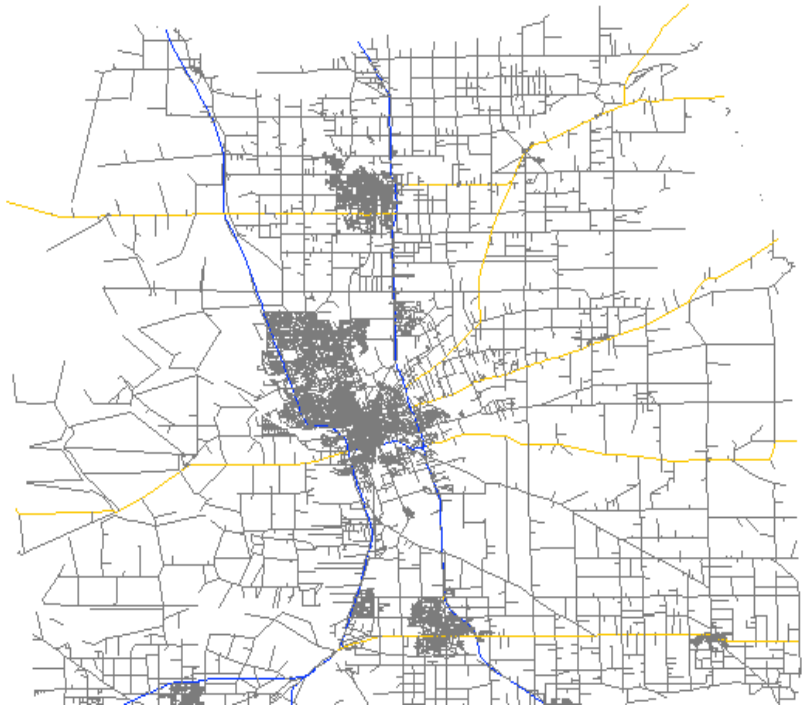
Chabini & Shan [Trans ITS'02]

# Time-dependent A* Search

- Goal:
  - Find a $h(v_i)$ that will never overestimate the time-dependent travel-time between $v_i$ and d. This is necessary for Exact results
  - $h(v_i)$ should be as close as possible to actual distances for Efficient processing of fastest path computation
- Approach:
  - **Step 1:** Partition the road network into non-overlapping partitions **(Offline)**
  - **Step 2:** Precompute $h(v_i)$ using distances in and between the non-overlapping partitions **(Offline)**

# Time-dependent A* Search

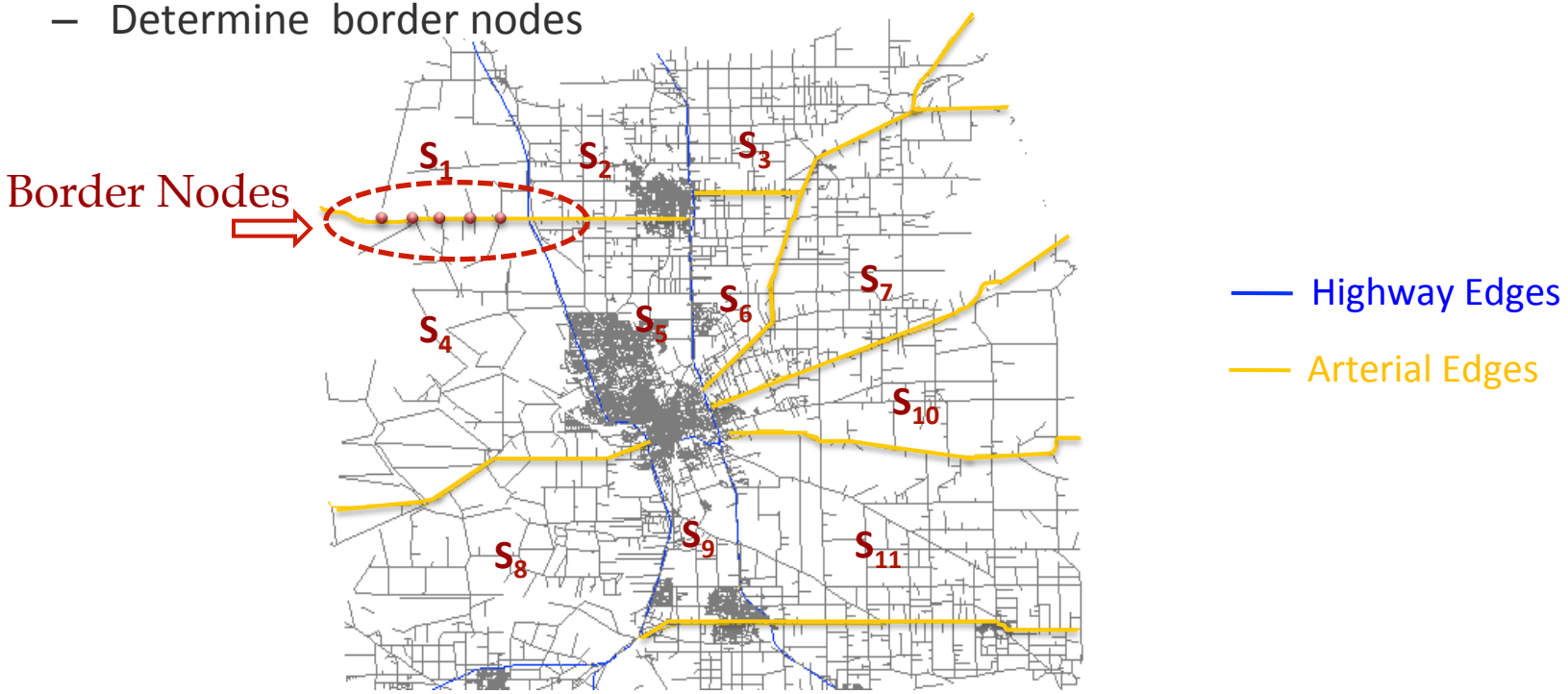
- **Step 1: Partition** the road network using network hierarchies
  - Partition the road network to highways (highest level)



Highway Edges

# Time-dependent A* Search

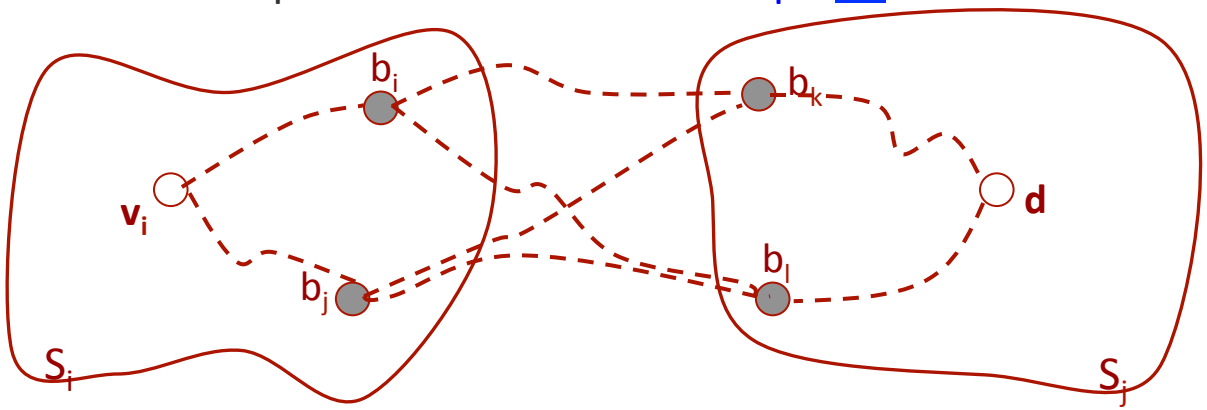
- **Step 1: Partition** the road network using network hierarchies
  - Partition the road network using highest level roads (i.e., highways)
  - Partition each partition using lover level road network (i.e., arterials)
  - Determine  border nodes




*Our algorithm yields correct results with all non-overlapping partitioning algorithms*

# Time-dependent A* Search

- **Step 2:** Compute **intra and inter distance labels**
  - **Intra:** fastest path in Lower-bound Graph G (where edge weights are travel-time, i.e., fastest speed) from each node $v_i$ to border nodes and border nodes to $v_i$
  - **Inter :** fastest path in Lower-bound Graph G between border nodes

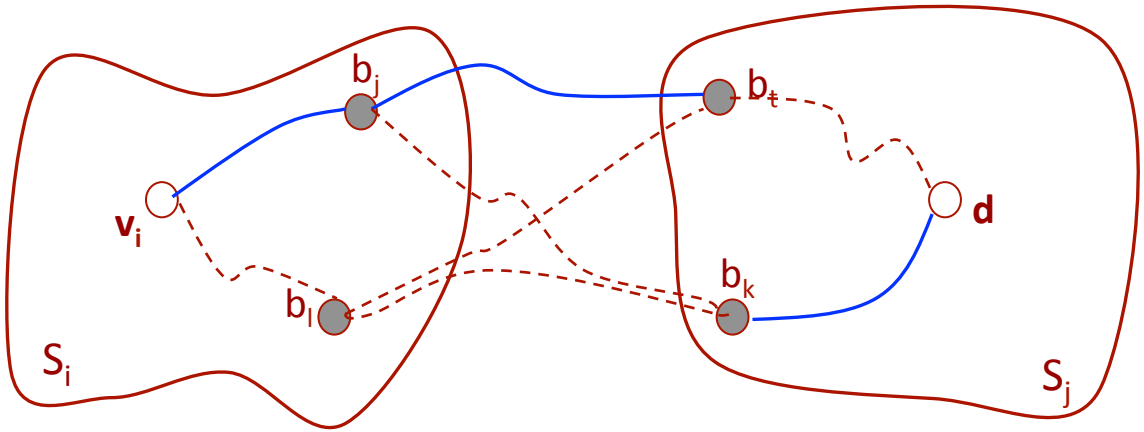



- Only store the minimum of node-to-border, border-to-border, and border-to-node travel times

$$LTT(v_i, b_i) = \arg\min(LTT(v_i, b_i), LTT(v_i, b_j))$$
$$LTT(b_l, d) = \arg\min(LTT(b_k, d), LTT(b_l, d))$$
$$LTT(b_i, b_k) = \arg\min(LTT(b_i, b_k), LTT(b_i, b_l), LTT(b_j, b_k), LTT(b_j, b_l))$$

# Time-dependent A* Search

- **Lemma:** $h(v_i,d)$ based on intra and inter distance labels is lower-bound of $TDFP(v_i,d,t)$:




- **Proof:** $h(v_i,d) <= TDFP(v_i,d,t_{vi})$

$$LTT(v_i,b_i) \le TDFP(v_i,b_i,t_{vi}),\ LTT(b_i,b_t) \le TDFP(b_i,b_t,t_{bi}),$$
$$LTT(b_k,d) \le TDFP(b_k,d,t_{bk})$$
$$h(v_i,d) = LTT(v_i,b_i) + LTT(b_i,b_t) + LTT(b_k,d) \le TDFP(v_i,d,t_{vi})$$

# Time-dependent A* Search

- **Low Storage Overhead**
  - Only partition, node-to-border and border-to-node information is added to each node $v_i$
  - Border-to-border information is a small fraction of the all network

| Node | Partition | Node-to-Border | Border-to-Node |
|---|---|---|---|
| $n_1$ | $S_1$ | $b_1$,5 | $b_1$,7 |
| $n_2$ | $S_1$ | $b_2$,6 | $b_3$,4 |
| …. | …. | …. | …. |
| $n_{41}$ | $S_9$ | $b_{17}$,3 | $b_{15}$,6 |
| $n_n$ | $S_k$ | $b_{u,}$,x | $b_v$,y |

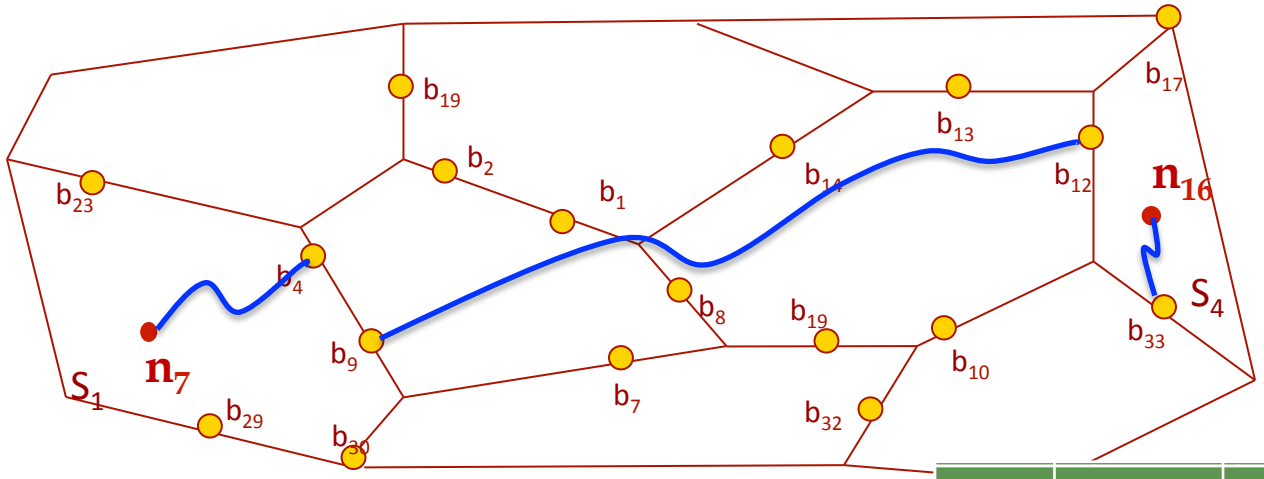
Node-to-Border (Intra)

| Border | Border | Distance | Partition |
|---|---|---|---|
| $b_1$ | $b_3$ | 14 | $S_{1,}S_4$ |
| $b_1$ | $b_{41}$ | 18 | $S_{1,}$S3 |
| $b_1$ | $b_{15}$ | 12 | $S_{4,}S_1$ |
| …. | …. | …. | |
| $b_n$ | $b_k$ | ….. | |

Border-to-Border (Inter)

# Time-dependent A* Search

- **Fast $h(v_i,d)$ computation**
  - $h(v_i,d)$ is computed by simple table look-ups (nanoseconds)




- **Efficient updates** $h(n_7,d) = 6+18+5$
  - Distance labels are only updated if lower-bound distances changed

| Node | Partition | Node-to-Border | Border-to-Node |
|---|---|---|---|
| $n_6$ | $S_1$ | $b_{23},5$ | $b_{23},7$ |
| $n_7$ | $S_1$ | $b_4,6$ | $b_9,4$ |
| …. | …. | …. | …. |
| **$n_{16}$** | **$S_4$** | $b_{17},3$ | **$b_{33},5$** |
| $n_n$ | $S_k$ | $b_{u,},x$ | $b_v,y$ |

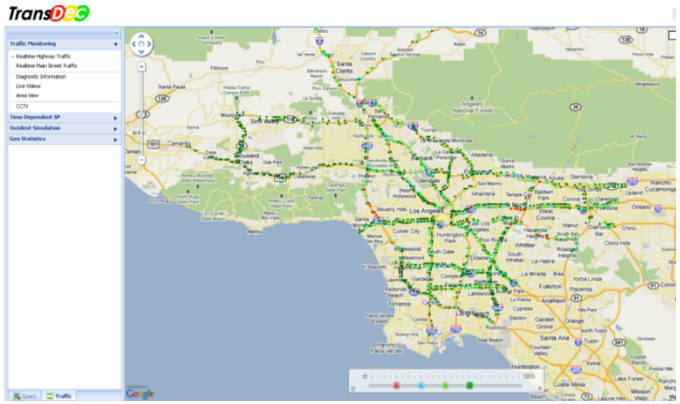
# Outline

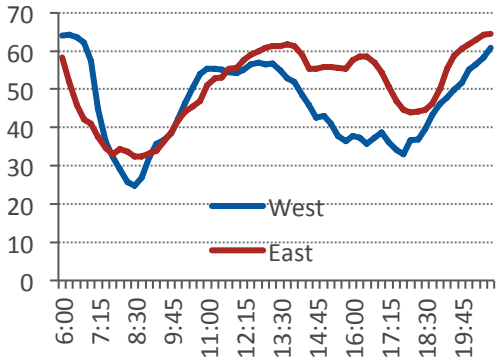
- Distance Computation
- Motivation
- Related Work
- Time-dependent A* Search
- Experimental Evaluation

# Experimental Evaluation

- **Road Network Dataset (obtained from Navteq)**
  - Los Angeles (LA) Network with 304,162 nodes
  - California (CA) Network with 1,965,300 nodes

- **Time-dependent Network Data (obtained from ADMS)**
  - LA Metro, Price School of Public Policy and IMSC
  - 9300 Sensors on freeways and arterials in LA
    - 1 sensor/reading per minute
    - Collecting and archiving past 2 years

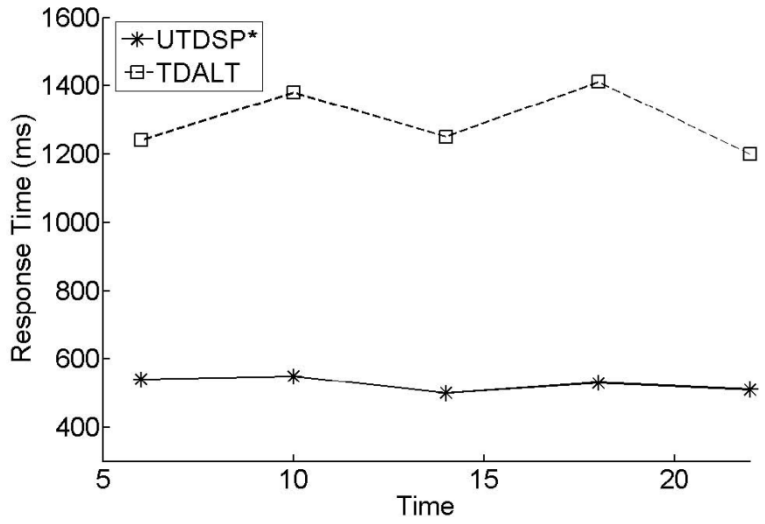



- **Experimental Setup:**
  - A server with 2.7 GHz Pent. Duo Core Proc. and 12GB RAM
  - Source, destination and departure time $t_s$ are determined uniformly at random
  - Average results computed from 1000 random s-d queries

# Experimental Evaluation

- **Comparison with TD-ALT**
  - TD-ALT: Determine 64 landmarks based on maxCover (best known landmark selection algorithm)
  - TDFP: Divide CA network to 64 partitions




*Derived from 1000 random s-d queries*

**Response Time:**

– TD-ALT very loose bounds based on the randomly selected *s and d, and hence the large search* space.

**Storage:**

– TD-ALT attaches each node an array of 64 elements. Total Storage = 63 MB for CA

– TDFP attaches each node an array of 2 elements (intra distance labels) and b-to-b. Total Storage=8.5 MB for CA

# Conclusion

**Research**

- Accurate traffic prediction (ICDM'12)
- Fastest-Path computation in time-dependent networks (SSTD'11)
- kNN search computation in time-dependent networks (DEXA'10)

**System Development**

- TransDec (ICDE'2010)

**Tech-Transfer**

- March 2013

# Acknowledgement

***Traffic Congestion:***


Balan Sethu Raman, MS


Dan Fay, MS


Prof. Giuliano (School of Policy)


Kali K. Fogel
LA-Metro

**Kenneth Coleman**
Motorist Services Program Manager at LA-Metro

***TransDec:***


Ugur Demiryurek


Barak Fishbain


Keivan Hamedaniraja


Afsin Akdogan


Colin Gu


Mohammad Ali, MS

***Research:***


Penny Pan


F. Banaei-Kashani


A. Ranganathan, IBM


Chetan Gupta, HP Labs

***ClearPath:***


Hamid Heidary, CEO


CTO


Chris O'Connell, VP Bus Dev


Phil Spivey, Board Member