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 inductance decreases when a car is on top of it.
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 = ($N\times L)/O$ where $L$ is a constant proportional to the average length of a car
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/
<table>
<thead>
<tr>
<th>Data Type</th>
<th>Sample Size (KB)</th>
<th>Hourly (in KB)</th>
<th>Daily (in KB)</th>
<th>Annual (in KB)</th>
<th>3 Years (in KB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>bus_mta_inv2.xml</td>
<td>23</td>
<td>0.96</td>
<td>23.00</td>
<td>8,395.00</td>
<td>25,185.00</td>
</tr>
<tr>
<td>bus_mta_rt2.xml</td>
<td>1065</td>
<td>532.50</td>
<td>31,950.00</td>
<td>279,882,000.00</td>
<td>839,646,000.00</td>
</tr>
<tr>
<td>cctv_inv.xml</td>
<td>57</td>
<td>0.04</td>
<td>2.38</td>
<td>20,805.00</td>
<td>62,415.00</td>
</tr>
<tr>
<td>cms_inv.xml</td>
<td>52</td>
<td>0.04</td>
<td>2.17</td>
<td>18,980.00</td>
<td>56,940.00</td>
</tr>
<tr>
<td>cms_rt.xml</td>
<td>48</td>
<td>38.40</td>
<td>2,304.00</td>
<td>20,183,040.00</td>
<td>60,549,120.00</td>
</tr>
<tr>
<td>event_d7.xml</td>
<td>11</td>
<td>8.80</td>
<td>528.00</td>
<td>4,625,280.00</td>
<td>13,875,840.00</td>
</tr>
<tr>
<td>rail_mta_inv.xml</td>
<td>1</td>
<td>0.00</td>
<td>0.04</td>
<td>365.00</td>
<td>1,095.00</td>
</tr>
<tr>
<td>rail_rt.xml</td>
<td>8</td>
<td>8.00</td>
<td>480.00</td>
<td>4,204,800.00</td>
<td>12,614,400.00</td>
</tr>
<tr>
<td>rms_inv.xml</td>
<td>865</td>
<td>0.60</td>
<td>36.04</td>
<td>947,175.00</td>
<td></td>
</tr>
<tr>
<td>rms_rt.xml</td>
<td>1236</td>
<td>988.80</td>
<td>59,328.00</td>
<td>519,713,280.00</td>
<td>1,559,139,840.00</td>
</tr>
<tr>
<td>signal_inv.xml</td>
<td>2095</td>
<td>1.45</td>
<td>87.29</td>
<td>764,675.00</td>
<td>2,294,025.00</td>
</tr>
<tr>
<td>signal_rt.xml</td>
<td>2636</td>
<td>3,514.67</td>
<td>210,880.00</td>
<td>1,847,308,800.00</td>
<td>5,541,926,400.00</td>
</tr>
<tr>
<td>tt_d7_inv.xml</td>
<td>746</td>
<td>0.52</td>
<td>31.08</td>
<td>272,290.00</td>
<td>816,870.00</td>
</tr>
<tr>
<td>tt_d7_rt.xml</td>
<td>152</td>
<td>152.00</td>
<td>9,120.00</td>
<td>79,891,200.00</td>
<td>239,673,600.00</td>
</tr>
<tr>
<td>vds_art_d7_inv.xml</td>
<td>115</td>
<td>0.08</td>
<td>4.79</td>
<td>115.00</td>
<td>41,975.00</td>
</tr>
<tr>
<td>vds_art_d7_rt.xml</td>
<td>45</td>
<td>45.00</td>
<td>2,700.00</td>
<td>64,800.00</td>
<td>23,652,000.00</td>
</tr>
<tr>
<td>vds_art_ladot_inv.xml</td>
<td>2538</td>
<td>1.76</td>
<td>105.75</td>
<td>926,370.00</td>
<td>2,779,110.00</td>
</tr>
<tr>
<td>vds_art_ladot_rt.xml</td>
<td>969</td>
<td>969.00</td>
<td>58,140.00</td>
<td>509,306,400.00</td>
<td>1,527,919,200.00</td>
</tr>
<tr>
<td>vds_fr_d7_inv.xml</td>
<td>957</td>
<td>0.66</td>
<td>39.88</td>
<td>957.00</td>
<td>349,305.00</td>
</tr>
<tr>
<td>vds_fr_d7_rt.xml</td>
<td>361</td>
<td>722.00</td>
<td>43,320.00</td>
<td>379,483,200.00</td>
<td>1,138,449,600.00</td>
</tr>
<tr>
<td><strong>Total KB from XML data</strong></td>
<td><strong>13980</strong></td>
<td><strong>864660</strong></td>
<td><strong>6,985.28</strong></td>
<td><strong>10,057,449.00</strong></td>
<td><strong>36,709,688.00</strong></td>
</tr>
</tbody>
</table>

Variety (gps, video, loop sensor, events)

Velocity

Volume

An Exclusive Contract w LA-Metro

A BIGDATA Problem: V³
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**

- **Highway** (4313)
- **Arterial** (4780)
- **Bus & Rail** (2000)
- **Ramp meter**
- **Events & CMS** (800/day)

**46 MB/min**

**Real-time Queries & Data Cleansing**

**26 TB/Year**

**Spatiotemporal Indexing**

*(Oracle Award, IEEE CloudCom Best paper)*

**E.g., Accident impact analysis & prediction**

*(ICDM’12)*

Intel

Oracle

Microsoft

IMSC

Integrated Media Systems Center

NSF

USC

School of Engineering
Product: **ClearPath**

**Main Differentiator: Predictive Path Planning**
Predictive vs. Real-Time Path-Planning

Best Route based on current conditions

7:10AM
Predictive vs. Real-Time Path-Planning

Evolution of traffic over time

7:15AM
Predictive vs. Real-Time Path-Planning

Hindsight: slower route

Hindsight: faster route

7:20AM
Comparisons (Better Path)

Venice ➔ USC

8:30 AM
ClearPath: 20 min
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)
Denver → USC

6:00 AM

6:45 AM

8:15 AM

10:00 AM

Comparisons (Path Alternatives)
Anaheim → USC

6:00 AM

6:45 AM

8:15 AM

10:00 AM
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 (2003-2010)

Spatial Network (2010-)

Euclidean Space

Edge weights change with time

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 $w_{12}(t), w_{34}(t)$
  – 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

Spatial Database

Shortest Path

Static Road Network TD-Road Network

• 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

Spatial Database

Shortest Path

Static Road Network

TD-Road Network

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

Precomputation:
- 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]

Inefficient: high storage cost and long precomputation time
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 Algorithm

A* Algorithm

Dijkstra: since (S,v_j) < (S,v_i), expand v_j first
A*: since (s,v_j)+h(v_j) < (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 \( h(v_i, d) \leq D(v_i, d, t) \) 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:

\[
D_{EUC}(v_i, d) = \frac{D_{EUC}(v_i, d)}{\max(speed)}
\]

- 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)
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 lower 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

\[
\begin{align*}
LTT(v_i, b_i) &= \arg \min(LTT(v_i, b_i), LTT(v_i, b_j)) \\
LTT(b_l, d) &= \arg \min(LTT(b_k, d), LTT(b_j, d)) \\
LTT(b_i, b_k) &= \arg \min(LTT(b_i, b_k), LTT(b_i, b_l), LTT(b_j, b_k), LTT(b_j, b_l))
\end{align*}
\]
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:** $h(v_i,d) \leq TDFP(v_i,d,t_{vi})$

  
  \[
  LTT(v_i, b_i) \leq TDFP(v_i, b_i, t_{vi}), \quad LTT(b_i, b_t) \leq TDFP(b_i, b_t, t_{bi}),
  \]
  \[
  LTT(b_k, d) \leq TDFP(b_k, d, t_{bk})
  \]

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

<table>
<thead>
<tr>
<th>Node</th>
<th>Partition</th>
<th>Node-to-Border</th>
<th>Border-to-Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n_1$</td>
<td>$S_1$</td>
<td>$b_{1,5}$</td>
<td>$b_{1,7}$</td>
</tr>
<tr>
<td>$n_2$</td>
<td>$S_1$</td>
<td>$b_{2,6}$</td>
<td>$b_{3,4}$</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
<td>....</td>
<td>....</td>
</tr>
<tr>
<td>$n_{41}$</td>
<td>$S_9$</td>
<td>$b_{17,3}$</td>
<td>$b_{15,6}$</td>
</tr>
<tr>
<td>$n_n$</td>
<td>$S_k$</td>
<td>$b_{u,x}$</td>
<td>$b_{v,y}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Border</th>
<th>Border</th>
<th>Distance</th>
<th>Partition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_1$</td>
<td>$b_3$</td>
<td>14</td>
<td>$S_1, S_4$</td>
</tr>
<tr>
<td>$b_1$</td>
<td>$b_{41}$</td>
<td>18</td>
<td>$S_1, S_3$</td>
</tr>
<tr>
<td>$b_1$</td>
<td>$b_{15}$</td>
<td>12</td>
<td>$S_4, S_1$</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
<td>....</td>
<td></td>
</tr>
<tr>
<td>$b_n$</td>
<td>$b_k$</td>
<td>.....</td>
<td></td>
</tr>
</tbody>
</table>

**Node-to-Border (Intra)**

**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)

![Diagram of a network with nodes and edges](image)

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

<table>
<thead>
<tr>
<th>Node</th>
<th>Partition</th>
<th>Node-to-Border</th>
<th>Border-to-Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n_6$</td>
<td>$S_1$</td>
<td>$b_{23},5$</td>
<td>$b_{23},7$</td>
</tr>
<tr>
<td>$n_7$</td>
<td>$S_1$</td>
<td>$b_{4},6$</td>
<td>$b_{9},4$</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>$n_{16}$</td>
<td>$S_4$</td>
<td>$b_{17},3$</td>
<td>$b_{33},5$</td>
</tr>
<tr>
<td>$n_n$</td>
<td>$S_k$</td>
<td>$b_{u},x$</td>
<td>$b_{v},y$</td>
</tr>
</tbody>
</table>
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

\[\begin{align*}
\text{West} & \quad \text{East} \\
6:00 & \quad 10 \\
7:15 & \quad 9 \quad \bullet \quad 12 \\
8:30 & \quad 11 \\
9:45 & \quad 10 \\
11:00 & \quad 9 \quad \bullet \quad 12 \\
12:15 & \quad 11 \\
13:30 & \quad 10 \\
14:45 & \quad 9 \quad \bullet \quad 12 \\
16:00 & \quad 11 \\
17:15 & \quad 10 \\
18:30 & \quad 9 \quad \bullet \quad 12 \\
19:45 & \quad 11 \\
\end{align*}\]
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

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

*Derived from 1000 random s-d queries*
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:

Kenneth Coleman
Motorist Services Program Manager at LA-Metro

Kali K. Fogel
LA-Metro

Balan Sethu Raman, MS

Dan Fay, MS

Prof. Giuliano (School of Policy)

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

IMSC
Integrated Media Systems Center

USC
School of Engineering