Monitoring Travel Time Reliability from the Cloud: Cloud Computing Based Architecture for Advanced Traffic Information Dissemination

Hao Lei
Department of Civil and Environmental Engineering
University of Utah
Salt Lake City, Utah, 84112
Email: hlei@eng.utah.edu

Tao Xing
Department of Civil and Environmental Engineering
University of Utah
Salt Lake City, Utah, 84112
Email: tao.xing@utah.edu

Xuesong Zhou*
Assistant Professor
Department of Civil and Environmental Engineering
University of Utah
Salt Lake City, Utah, 84112
Tel: 801-585-6590 Fax: 801-585-5477
Email: zhou@eng.utah.edu

July 2010

Submitted for consideration for presentation and publication at the 90th Annual Meeting of the Transportation Research Board, January 23-27, 2010

Text=4950
Figures (8 x 250) = 2000
Total words = 6950

* Corresponding Author
ABSTRACT

Under the existing loosely-distributed sensing environment with heterogeneous data sources, transportation planning and management agencies have found a critical need for efficiently storing, processing and extracting desirable corridor-level and network-level information through a systematic and seamless integration of data sources. Extracting travel time variability and trip reliability information from a large amount of spatially correlated data from a large-scale network with dramatic variations in travel demand and road capacity creates another demanding challenge to traffic engineers. The emerging practice of Cloud Computing provides a revolutionary solution platform to meet the above mentioned needs. This paper presents a cloud computing based system architecture to provide unified data storage and computing platform for managing a large volume of data. An Ensemble Kalman filtering based data mining and fusion model is further developed to integrate heterogeneous sources of sensor data. The traveler information provision system provides a web and mobile phone based user interface for providing a convenient user interaction. Test results demonstrate that the applicability of cloud computing technique in such data-rich and compute intensive applications.
1. INTRODUCTION

Over the last few decades, federal, state and local agencies have deployed many traffic sensor systems to monitor and manage freeway and arterial networks. With these installed sensors, for example, widely-used inductive loop detectors, valuable traffic measurements are collected, processed and further disseminated to travelers and system managers to make informed decisions. In recent years, emerging techniques, such as Automatic Vehicle Identification (AVI) and Global Positioning Systems (GPS) have been widely incorporated into network monitoring, signal control, toll collection and other traffic applications. In the present, loosely-distributed sensing environment with heterogeneous data sources, transportation planning and management agencies have found a critical need for efficiently storing, processing and extracting desirable corridor-level and network-level information through a systematic and seamless integration of data sources.

In transportation planning and operation applications, travel time reliability is an essential system performance index and provides a critical Level of Service (LOS) measure of the transportation network, in additional to regular mobility measures such as travel time indices and congestion levels. Trip reliability information, such as the on-time arrival likelihood of traveling alternatives/options, can also help travelers better schedule their departure time, mode and routes to reduce the unexpected time or monetary loss due to recurring or non-recurring congestion. In a large-scale network with dramatic variations in travel demand and road capacity, it is a complex task to extract travel time variability and trip reliability information from a large amount of spatially correlated data. Furthermore, it is also computationally challenging to compute the most reliable path for travelers to optimize their routes in real-time.

Thanks to the advances of telecommunication and information technologies, the emerging practice of Cloud Computing provides a revolutionary solution platform to meet the above-mentioned needs. It represents a fundamentally new approach and opportunities for large-scale system monitoring, data archiving, and information fusion in transportation planning and operations. In a cloud computing environment, the computational power and data storage are integrated in a unified platform. The utility computing model employed by cloud computing service providers dramatically reduces the cost and complexity of managing computer servers, datacenters and communication networks. The utility computing model is analogous to how traditional utility services, i.e. water, electricity, are consumed. This provides cloud computing users with the ability to choose customized computational power and storage space and pay for only what they consume. The Internet-based nature of cloud computing enables distributed storage, universally-accessible data and flexible client-server application implementations. In essence, cloud computing provides a heterogeneous, robust and open distributed computing environment.

In this paper, a travel time reliability-oriented traffic information provision system is designed and implemented to demonstrate the applicability of the cloud computing technique, particularly, by addressing the following three key challenges.

- Heterogeneous Data Sources

Due to their associated low unit equipment cost and relatively high performance, inductive loops have become the predominant fixed vehicle detection device in the United States. Currently, Radio Frequency Identification (RFID) -based Automatic Vehicle Identification (AVI) systems are widely deployed in road pricing, parking lot management, as well as real-time travel time information provision. In past several years, in-car navigation using Global Positioning System (GPS) technology has matured into a rapidly growing industry and its penetration rate in the U.S. is expected to exceed 10% in 2010. Data from different sources (that is, point, point-to-point, semi-continuous data) usually requires different processing techniques. Thus, an ability to integrate valuable travel time reliability information in this heterogeneous data environment is critically needed.
Besides the heterogeneity of the data sources, the volume of the data received every day is a considerably large amount for storing and processing. For instance, the Southern California Region (covering Los Angeles and Orange Counties) with 5.3 Megabytes households and 7.1 Megabytes vehicles generate about 41.2 Megabytes motorized and non-motorized trips every day. If the location data (in terms of longitude and latitude) are recorded every second for each trip (assuming 30 min. trips), about 0.6 TB of data will be generated in a single day. As a result, methods for cost-effective and computationally-efficient storage and further processing for these large data volumes must be carefully examined and planned, especially for regional real-time travel information provision applications.

If without travel time reliability information, the accuracy of the predicted travel time for a trip is of limited values. Therefore, an effective travel information provision system must provide reliability related routing services. Travel time variability statistics can better represent a commuter’s experience than a simple average trip time. There are a wide range of travel time reliability measures, and most measures compare extreme-delay days to those with an average delay. The most frequently adopted definitions of measuring travel time reliability are 90th or 95th percentile travel times, buffer index, and planning time index. Taking the definition of 90th or 95th percentile travel times as an example, it estimates how bad the delay will be on specific routes during the heaviest traffic days. Specifically, the one or two worst days of each month usually mark the 95th or 90th percentile. In addition, several statistical measures, such as the standard deviation and coefficient of variation, have also been used to quantify the travel time reliability. Although not easy for a nontechnical audience to understand, the standard deviation has been calibrated in several empirical studies for route choice utility functions under stochastic traffic conditions (10, 11, and 13).

In reality, travel times among different links could be highly correlated, e.g. due to the propagation of congestion from a downstream bottleneck to an upstream link along a freeway or arterial corridor. Figure 1 shows that strong correlations exist among link travel times on a corridor of Bayshore Freeway in California.

**FIGURE 1.** Link travel time correlation on a corridor of Bayshore Freeway, CA.
Recently, spatial and temporal correlations among links have been exclusively studied by a number of researchers. Miller-Hooks and Mahmassani (7) and Nie and Wu (8) have contributed substantial research efforts on the a priori time-varying least travel time problem. In the study of the nonlinear disutility shortest path problem by Boyles and Waller (2) and the reliable routing problem by Nie and Wu (9), limited spatial and temporal dependences have been considered. In their studies, the randomness of link travel time is modeled by probability density functions abstracted from the historical database. Meanwhile, limited spatial correlation is captured through a Markovian model that considers the transition probabilities of link states.

Calculating the measure of travel time reliability is a data-intensive computational task. In order to capture the variability of path travel time, a sufficiently-large amount of the historical travel time records are required over multiple days or months. Additionally, for the standard deviation-based travel time reliability calculation used in this study, e.g. \( \min \text{ mean} + \beta \sqrt{\text{var}} \), the nonlinear and nonadditive characteristic introduces additional numerical difficulty compared to standard routing problems.

2. SYSTEM ARCHITECTURE AND DATA FLOW

In this study, we design a travel time reliability-oriented traffic monitoring and information provision system based on a cloud computing platform. This Internet-based service is implemented as a generic client-server model. The system architecture is shown in Figure 2, and section 2.1 introduces the major modules and data flow among them; Section 2.2 briefly presents the client design considerations.

![Cloud computing-supported system architecture](image.png)

**FIGURE 2. Cloud computing-supported system architecture.**

2.1 Server-side components

As shown in Figure 2, the server is composed of four core components:

- Traffic measurement and historical pattern databases
- GPS map matching engine
- Data mining and fusion engine
Reliable routing engine

**Databases**
It is desirable to store a large volume of loop detector, AVI and GPS trace data in distributed rational databases. The physical locations of these datacenters are transparent to the users and traffic system managers. Possible accessing interfaces, accordingly, must be provided to allow seamless information sharing among commuters, public-sector agencies and private traveler information providers. A cloud computing database system offers new opportunities for potential public-private partnerships in sharing and trading different data streams.

**GPS map matching engine**
Map matching is an essential data processing service that converts raw GPS location data samples to useful traffic information in node-link traffic network representation form. User-provided GPS trajectories are processed in this module and fed into the subsequent data mining and fusion tasks.

**Data mining and fusion engine**
This study develops an Ensemble Kalman filtering-based data mining and fusion module, so as to extract and integrate valuable end-to-end trip time variability information from various sources.

**Reliable Routing Engine**
As the building block of the entire traffic information provision system, the reliable routing engine calculates routes under different criteria, based on live traffic data from the traffic data fusion and prediction engine, and further generates final route guidance information to end users.

2.2 Client-side components
On the client side, two software packages have been developed to fulfill the needs of both commuters and traffic system operators. As shown in Figure 3(b), the web-based application works as a network-wide traffic analysis tool for system operators and a pre-trip planner for commuters. Specifically, as a traffic analysis tool, it visualizes path-level traffic estimation/prediction results and allows users to conduct temporal/spatial sensitivity analysis. As a pre-trip planner, it offers commuters a variety of options for route, travel mode and departure time choices. For smart-phone users, a Windows Mobile-based application, shown in Figure 3(a) is developed to provide customized real-time route guidance and up-to-date traffic information.

FIGURE 3 GPS-enabled smart phone and web-based interfaces for disseminating travel time reliability information.
In this section, we first introduce how general Kalman filtering methods can be adapted to estimate the link travel time based on existing data and update the estimated link travel time with newly received measurements. Section 3.2 elaborates how the sample-based representation mechanism provided by Ensemble Kalman filtering achieves potential computational efficiency improvements while still maintaining near-optimal estimation quality.

### 3.1 Kalman Filtering Framework Handling Multiple Data Sources

The Kalman Filter is a sequential filter method, in which the system model is updated forward in time. Whenever new measurements are available, these are used to reinitialize the model before the integration continues. In the proposed system, the Kalman filtering method acts as a data mining and fusing engine due to its capability of processing heterogeneous sources of data. It is adapted to estimate the link travel time using existing data; with new data, predictions are made to reflect the new trend when traffic evolves.

The historical (a priori) and updated (a posteriori) estimates of the system at the time step $\theta$ are denoted as $T_\theta^-$ and $T_\theta^+$ respectively.

The updating equation is a linear combination of the measurements $D_\theta$ and the historical model state $T_\theta^-$. The linear combination is chosen to minimize the variance in the analyzed estimate $T_\theta^+$, which is given by the equation

$$T_\theta^+ = T_\theta^- + K_\theta(D_\theta - H_\theta T_\theta^-)$$

where $H$ is the measurement operator relating the true model state $T$ to the observations $D$ allowing for measurement errors $R$ ignoring the time index:

$$D = HT + R$$

where $H = \begin{bmatrix} h_{i1}, h_{i2}, \ldots, h_{ij} \\ h_{21}, h_{22}, \ldots, h_{2j} \\ \ldots \ldots \ldots \\ h_{11}, h_{12}, \ldots, h_{1j} \end{bmatrix}$ and $h_{ij} = \begin{cases} 1 & \text{link } j \text{ is included in observation } i \\ 0 & \text{otherwise} \end{cases}$

To process observations from heterogeneous sources, we must carefully construct the mapping matrix $H$, to relate different forms of measurements to the true link travel times.

Considering the example network shown in Figure 4, the true link travel times are denoted as $T = t_1, t_2, t_3$ where $t_1$, $t_2$, and $t_3$ are link travel times on links 1, 2, and 3, respectively. Suppose that point sensors are deployed on links 1 and 2. To map the point measurements $D' = d_1, d_2$ to the true state, the mapping matrix is defined as $H = \begin{bmatrix} 1,0,0 \\ 0,1,0 \end{bmatrix}$ so that $D' = HT + R'$, where $R'$ is the measurement errors associated with this specific measurement. Meanwhile, if there is also an AVI measurement $D^* = d_3$ which measures the travel time along path A-B-C, the mapping matrix has to be defined as $H = [1,1,0]$ so that $D^* = HT + R^*$ where $R^*$ is the error associated with this measurement. Similarly, we can construct a matrix $H$ to map GPS traces to the underlying network links, which is the main output of the GPS map matching algorithm.

![FIGURE 4 A simple example network.](image-url)
The predicting process is expressed by
\[ T_{\theta+1} = U T_{\theta} + w \] \hspace{1cm} (3)
where \( U \) represents the function to predict the system state at time step \( \theta + 1 \) from the state in the previous time step \( \theta \) and \( w \) is the system error. If \( U T_{\theta} = T_{\theta} \), the predicting process becomes a Moving Average (MA) process.

The respective covariance for prediction, analysis and measurements are denoted to be \( P^- \), \( P^+ \) and \( R \). And the Kalman gain matrix \( K \) is calculated by:
\[ K = P^+ H^T (HP^+ H^T + R)^{-1} \] \hspace{1cm} (4)
The analysis error covariance is updated as
\[ P^+ = P^- - KHP^- = (I - KH)P^- \] \hspace{1cm} (5)

### 3.2 Ensemble Kalman Filtering for Estimating Spatial Correlation and Travel Time Variability

To avoid high computational costs and accommodate nonlinearity in a dynamic system, ensemble Kalman filtering (EnKF) uses a sample-based representation to characterize the distribution of the system state and dramatically reduce the computational complexity while still providing suboptimal estimation of the system state and variance (4).

In EnKF, an ensemble \( \bar{X} \) of model states is represented by \( X = [X_1, ..., X_s, ..., X_S] \) where \( X_s \) is the \( s^{\text{th}} \) ensemble member and \( S \) is the number of the ensemble member. Each \( X_s \) is a vector, where
\[ \bar{X}_s = \begin{bmatrix} t_{s,1}, ..., t_{s,m}, ..., t_{s,M} \end{bmatrix}^T \]
and the \( m^{\text{th}} \) element \( t_{s,m} \) represents the state of the link \( m = [1,...,M] \) where \( M \) is the number of the link in the network. The ensemble size limits the number of degrees of freedom used to represent prediction and analysis errors and reduce the computational costs.

The ensemble mean \( \bar{X} \) is a vector, defined as \( \bar{X} = \bar{t}_1, ..., \bar{t}_m, ..., \bar{t}_M \). The \( m^{\text{th}} \) row of \( \bar{X} \) is the average link travel time of link \( m \) and is calculated by \( \bar{t}_m = \frac{1}{S-1} \sum_{s=1}^{S} t_{s,m} \).

The covariance matrix \( \bar{P} \) is calculated by using the relation:
\[ \bar{P} = \frac{1}{S-1} \sum_{s=1}^{S} (X_s - \bar{X})(X_s - \bar{X})^T \] \hspace{1cm} (6)

The Kalman gain matrix \( K \) in Eq. (6) is now obtained simply by replacing the state covariance by the covariance calculated from ensemble members:
\[ K = PH^T (HPH^T + R)^{-1} \] \hspace{1cm} (7)
Meanwhile, the ensemble is updated based on Eq. (4):
\[ X_s^+ = X_s^- + K(D - HX_s^-) \] \hspace{1cm} (8)
where \( D = [D_{s,1}, ..., D_{s,y}, ..., D_{s,y}] \), is a synthetic vector of perturbations of observations \( D_y \):
\[ D_{s,y} = d_y + \varepsilon_y \] \hspace{1cm} (9)
where \( \varepsilon_y \) is the added perturbation and its covariance is equal to \( R \). It has been recognized that without perturbing the observation, the analysis error covariance carried by the ensemble becomes less than the value given by the Kalman filter and therefore results in a premature reduction in the ensemble spread (3, 5).

As observed from Eq. (9) and (8), the perturbation of a new measurement for ensemble member updating and the updating process of each ensemble member are both an independent process. Therefore, the ensemble updating step can be decomposed into several sub-ensemble groups and processed in separate instances in the proposed cloud computing application.
The ensemble-based task decomposition and parallel updating process is illustrated in Figure 5. For example, a new raw AVI or GPS data sample must first be perturbed in the data perturbation module, and then a data dispatcher dispatches perturbed data to the corresponding ensemble update modules. The example ensemble set in this example has 10 ensemble members and is decomposed into two groups. Two ensemble updating modules simultaneously update the ensemble members inside each group. There are typically hundreds of ensemble members needed for reducing sampling errors, so this parallel updating mechanism can greatly improve the efficiency in a cloud computing-based distributed processing platform.

4. ROUTING ENGINE

In the proposed travel time reliability information provision system, the core routing engine calculates the most reliable path between origin and destination, specified by end users, and then provides the necessary route information to be visualized on mobile phones and web browsers. This section briefly presents the underlying algorithms and application-specific cost function design.

4.1 Core Routing Engine with Generalized Link Cost Function

In the proposed system, a computationally-efficient label-correcting shortest path algorithm with double-ended queue (deque) data structure is implemented in the routing engine. By accepting alternative link cost functions and underlying network representations, the routing engine is able to perform different application-specific computations. Furthermore, the object-oriented structure of the routing engine makes it suitable to be distributed as individual computational instances in the cloud computing system.
As an extension from the core routing engine, a route generation algorithm is developed for emerging ATIS applications such as Personal Navigation Devices (PND). It aims to provide a set of personalized alternative routes and corresponding predicted times of arrival to encourage smart route decisions. In this study, we apply a multi-criteria optimization framework to systematically consider and meet the route generation requirements of feasibility, attractiveness and representativeness. Specifically, link travel time attributes are considered as the first criterion, while the coverage of generated routes corresponding to the observation, or the diversion of generated routes is considered as the alternative criterion. By minimizing the overlap with respect to the least travel time route, the route generation algorithm finds alternative routes. The coverage diversity and repressiveness of alternative routes is achieved by dynamically penalizing the generalized link costs of previously generated routes. The route generation module provides a representative candidate pool for the route selection under variant user criteria, such as distance, travel time, economic characteristics, safety and multi-mode travel.

4.2 Algorithm for Finding Most Reliable Routes

Path travel time reliability is a critical measure of the quality of service for transportation systems and an important attribute in travelers’ route and departure time planning. Based on the routing engine, the proposed system adapts two lower bound algorithms for finding the most reliable path with and without link correlation, where the path travel time variability is represented by its standard deviation. Specifically, to handle the nonlinear and nonadditive cost functions introduced by the quadratic forms of the standard deviation term, a Lagrangian substitution approach is adopted to estimate the lower bound of the most reliable path solution through solving a sequence of standard shortest path problems. The Lagrangian substitution method reformulates the original problem as a linear and additive approximation dual problem. By solving the dual problem, we are able to determine a lower bound for each feasible solution and the corresponding solution quality. A subgradient algorithm is used to iteratively improve the solution quality by reducing the optimality gap. Interested readers are referred to the working paper by Xing and Zhou (2010) (14).

Furthermore, to characterize the link travel time correlation structure associated with the end-to-end travel time reliability measure, this research develops a sampling-based method to dynamically construct a proxy objective function in terms of travel time observations from multiple days. As mentioned before, link travel times can be highly correlated in a congested transportation network, due to the network structure and demand patterns. In order to explicitly include the link correlation in path travel time variable calculation, a sampling-based method adopted from the field of robust optimization is applied to approximate multi-day travel time records through a Monte Carlo approach. Specifically, the spatial correlation is incorporated through the travel time dependencies among measurement samples of different days.

As the shortest path calculation is still the most time-intensive component in finding the most reliable path, the most reliable route problem can be also naturally decomposed to different CPU cores of a cloud computing structure, by properly assigning the weighting parameters of the subgradient method. Moreover, within a sampling-based framework, instead of using a weighted combination of multi-day travel times as the link cost function, an alternative approximation is to fully utilize the resources in the cloud computing environment and find the most reliable path from the least expected travel time paths of each sample day.

4.3 Routing Engine for GPS Data Map-matching

Map matching is a core task in converting raw GPS location data to useful traffic information. An offline map matching module is equipped in the proposed system with a time-dependent shortest path algorithm-based approach that can simultaneously estimate the most likely activity tour on a transportation network and the resulting travel time on matched links. Different from traditional methods using only the geometric or topological information of the network, the proposed map matching module fully leverages the advantage of the routing engine and the temporal information of the GPS trajectory. In particular, the algorithm applied here utilizes a time-dependent, curve-to-curve
likelihood function for link costs and simultaneously updates the link travel times when determining the link trace of the GPS records. Moreover, the computational efficiency of the map matching process can be improved by specifying a sub-network for a particular GPS trajectory. With a distributed computing structure, GPS records can be parallelly processed in different machines/CPUs and update the historical travel time database simultaneously.

5. PLATFORM SELECTION

The benefits of selecting cloud computing include the savings in upfront capital expenditures on hardware and software and low management overhead. All the services are scalable and customized (1), and users decide the initial storage space and computational power their applications require. Later, users can dynamically adjust their needs whenever the applications require more or less space or power.

In Azure, the cloud computing platform provided by Microsoft, the compute service is built from one or more roles. A role defines a component that may run in the execution environment; within Windows Azure, a service may run one or more instances of a role. The number of instances is configurable by the users so that when needs evolve, the users can promptly adjust the instance number to flexibly accommodate new needs.

Furthermore, the cloud computing platform has already addressed many vital performance and security issues faced by common web application designs. For example, load balancing is a decisive issue related to application performance, and its implementation requires specific expertise and experience, of which smaller organizations and agencies often lack. Fortunately, many cloud computing platforms have implemented built-in load balancers which help to preserve users’ precious time and resources so that they can focus on business delivery. Prominent cloud computing service providers, such as Microsoft, Amazon and Google, have strong technical and supportive teams and can provide quick responses to potential security issues, thus assuring system security and integrity. All these advantages make cloud computing a very appropriate, reliable information provision platform for agencies. Its open characteristics also encourage information sharing with other private traveler information providers.

Several factors leading to our selection of Microsoft Azure for our system implementation are listed as follows:

- Full compatible to the existing development tools (Microsoft Visual Studio 2008)
- Easy debugging – Develop and debug locally before deployment with Visual Studio
- Easy deployment – Deployment tools integrated with Visual Studio
- Minimized porting efforts – Existing tools and applications are developed with Visual Studio

6. SYSTEM PERFORMANCE

In order to demonstrate the applicability of cloud computing on travel time reliability information provision service, a system performance test is conducted on a large-scale real-world transportation network at Bay Area, CA.

In Section 6.1, we briefly introduce the test environments, including the transportation network and server configuration adopted in the following system performance test. Section 6.2 presents the methods designed for the test. The test results are provided in Section 6.3.

6.1 Test Environment

The test bed for the proposed cloud computing platform-based travel time reliability information provision system is Bay Area, CA, which comprises of 53,124 nodes and 93,900 links.
The routing engine is coded in C# and deployed on the Microsoft Azure platform. The instance of routing engine is equipped with a 1.6 GHz CPU and 1.75 gigabytes of memory, the smallest compute instance size of the four unique sizes provided by the Microsoft Azure platform.

6.2 Test Methods and Tools

Two key system performance indices of the traveler information provision service are (1) the average number of simultaneous user requests the server can handle per unit time and (2) the average response time per request. The incoming user request rate is assumed to follow a Poisson distribution to reflect the randomness of the user requests. User requests are simulated by Oracle Load Testing for Web Application (OLTWA) (12). Detailed performance test result figures are generated by OLTWA.

6.3 Test Results

The test result in Figure 7 shows that, after 3 or 4 minutes of system warming-up, the number of requests that the server being tested can handle oscillates around 25 to 30 requests per second. Typically, if a user sends a request every 30 seconds, the computational instance capacity is 750 to 900 users per user request cycle.
The average response time for receiving and processing a user request is 1.23 seconds, as shown in Figure 8, which means the reliability information can be calculated by the proposed system in an acceptable time.

7. CONCLUDING REMARKS

Real-time traveler information provision service is a data-rich, computation-intensive application. If travel time reliability information needs to be provided along with the mean travel time, this will lead to increased storage and computational power requirements. The emerging cloud computing technique provides a new approach for large-scale system modeling, data storage, and diverse data fusion and mining. Thus, the travel time reliability-oriented travel information provision system implemented on the Microsoft Azure cloud computing platform makes a good demonstration of the applicability of cloud computing in such traffic applications.
In this study, methodological and analytic enhancements to existing travel time estimation and prediction models have been proposed for the purpose of capturing the spatial and temporal correlation of link travel times in real-world transportation networks and increasing prediction accuracy. Particular attention is focused on how to incorporate the revolutionary information technique to solve the challenges of large-scale real-time traveler information provision applications. The study described in this paper provides the following contributions to existing models and implements:

1. A cloud computing based system architecture which provides unified data storage and computing platform to manage and manipulate a large volume of data.
2. An Ensemble Kalman filtering based data mining and fusion model which integrates heterogeneous sources of sensor data.
3. An effective routing engine which provides efficient and accurate calculations of travel time reliability information on a large-scale real-world network.
4. A web and mobile phone based user interface providing a convenient user interaction.
ACKNOWLEDGEMENT

This paper is based on research work partially supported through the Safe Trip 21 project, sponsored by the Research and Innovative Technology Administration (RITA), US Department of Transportation. We would like to thank the California Partners for Advanced Transit and Highways (PATH) in the University of California, Berkeley, Safe Driving Systems, and NAVTEQ Corporate for providing traffic measurement and map data of the Bay Area, CA. We also want to thank former team members Sankhadip (Shawn) Sengupta, Claurissa Tuttle, Kenneth Williams, Rohan Madtha, Chao Yang and Daniel Van Tasell for their contributions on the development of the system. Special thanks to our colleague Mr. Jeffrey Taylor in the University of Utah for his careful review and comments. The work presented in this paper remains the sole responsibility of the authors.
REFERENCES