6U-08

A Cloud-based Architecture for Congestion Management

Rafik Hadfi† Yoshitaka Torii† Takayuki Ito†
Sho Tokuda† Ryo Kanamori‡ Masaaki Tanaka†
†Nagoya Institute of Technology ‡Nagoya University

1. Introduction

We propose a traffic and congestion management framework that uses a multi-agent architecture for dynamic routing and traffic prediction. It is a smart Cyber Physical Cloud [1] with all the requirements of any M2M formwork [2]. The smart aspect of the system relies on the usage of multi-agent technologies in the sense that we seek a platform that makes usage of smart prediction and consensus-making algorithms. In the present paper, we propose an overview of the architecture of the system, as well as the idea behind our preliminary multi-round dynamic traffic mechanism.

2. Architecture

2.1. System overview

The general overview of the system is shown in Figure 1. The architecture is divided into three main parts: the client tier, the application tier, and the data collection tier.

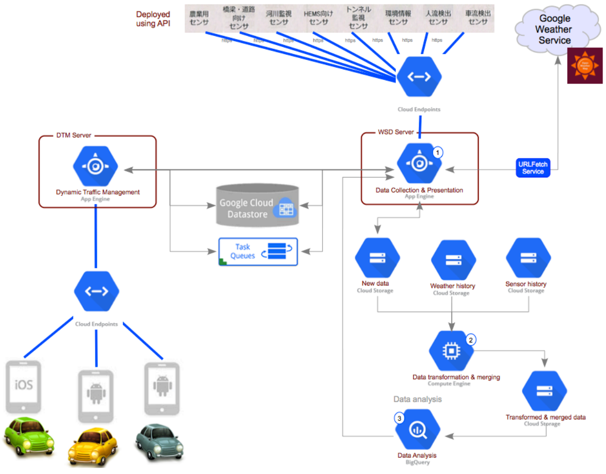

Figure 1. Architecture Overview

2.2. Client tier

This tier represents the direct users, defined as the individuals who interact with the system through their mobile phones (*Androids*) and seek guidance when navigating the streets in their cars. Their main task is to provide a path and get in turn an improved and optimal one based on the current traffic flow. In future versions of the system, users might provide their own predictions [5] of the traffic status as well as their fuel consumption.

2.3. Application tier

This is the business logic of the system holding the multi-agent algorithms and consensus-building mechanisms. This part implements the traffic management routines as well as any other implementation of smart routing mechanisms [5]. More details about our current routing scheme are provided in section 3.

2.4. Data collection tier

This is the part that handles the huge amount of data that needs to be collected, transformed, and stored in order to be used in the application tier. In fact, the data will be used in the prediction and the management of congestion using the traffic-related data as well as the weather data.

Herein, we assume that our sensors are distributed around a particular area (Japan), collecting different types of data.

The goal is to build a real-time dashboard of this data and enable an ad-hoc analysis of historical data merged with weather data. The result will be later used for the prediction of the traffic. The sensor data is gathered from several resources with a specific data format requirement. We distinguish several types of data sensors: agricultural, bridges, road, river monitoring, tunnel, environment information, crowd flow detection, traffic sensors, and so forth.

The sensor data will have to be collected in a highly scalable fashion, in order be transformed and analyzed using the *Google Cloud Platform*. Additionally, we have the weather data, accessible through remote calls, and provided using either *Google Weather* services, *OpenWeatherMap*, or any other proprietary service. The data collection is performed using a data pipeline based on the *Google App Engine*, and the *Google Cloud Endpoints* to describe and host our API endpoint. The APIs will be provided to the client-developers (Android for instance).

3. Traffic Management approach

We adopted a multi-round single-path minimum cost flow [4] routing scheme that attempts to route all the cars along several paths and thus exploits the whole network resources [3]. The paths chosen and followed by the cars are mapped onto a grid, where each point represents one street intersection, as in Figure 2 (intersections in red).

Figure 2. Four paths and the corresponding grid

The whole network is represented as a directed graph $G = (V, E)$ with a set V of vertexes (street intersections) and a set E of edges (links or roads). We assume that for each link $(i, j) \in E$ is assigned a capacity u_{ij}^t at time t, representing the maximum number of cars that can flow on the corresponding link. Similarly, each link $(i, j) \in E$ has a cost c_{ij}^t that corresponds to the cost per unit flow on that road. Most importantly, each vertex has a value b_i^t representing either the *supply* or the *demand*. If $b_i^t > 0$, the vertex i is a supply vertices, and if $b_i^t < 0$, the vertex i is a demand vertex. For a particular time t, the parameterized network could be described as $G(t) = (V, E, u^t, c^t, b^t)$.

If we represent the flow on arc $(i, j) \in E$ by x_{ij}^t we can obtain the optimization model for the minimum cost flow problem.

That is, at time t, and for car k, the objective function is (1), and is subject to (2) and (3).

$$x_k^t = \min_{x^t} \sum_{(i,j)\in E} c_{ij} x_{ij}^t \qquad (1)$$

$$\sum_{\{j:(i,j)\in E\}} x_{ij}^t - \sum_{\{j:(j,i)\in E\}} x_{ji}^t = b_i^t \ \forall i \in V \qquad (2)$$

$$x_{ij}^t \in \left[0, u_{ij}\right] \ \forall (i,j) \in E \qquad (3)$$

This minimum cost flow problem [4] can be solved using simple linear programming as to find the optimal flow for a given car k at time t. All cars are supposed to submit their paths as a sequence (4) of locations on the map.

$$p_k^t = [(x_1, y_1), \dots, (x_i, y_i), \dots, (x_n, y_n)] \qquad (4)$$

The algorithm will collect all the submitted paths, construct a network based on the locations as well as the existing roads of the grid, and then individually optimize for each car given the other cars' constraints (choices). This process is performed in an iterative manner until an eventual convergence. By convergence we refer to a distributed optimization of the overall cars' requests as to find the less congested routes all along the selected area. For instance, given 6 cars and their chosen paths, we can obtain the graphical representation in Figure 3 with the optimal flow for one single car.

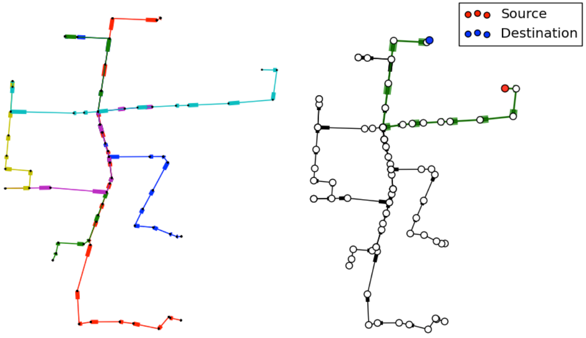

Figure 3. Paths and optimal route for one car

On the left side of Figure 3, we illustrate 6 paths in different colors corresponding to the cars' chosen routes for a particular time t. The network is mapped onto a grid of 85 intersections representing the search space of the algorithm that could eventually be enriched by adding more free routes. For one particular car, we get an optimal route having a flow equal to 12 within a route having a maximal capacity of 3 cars. The system will have to propose the route having the minimal cost flow to the concerned car. The car (driver) can either take the proposed route or propose another path. In future versions of the system we ought to incentivize the users to adopt the system and follow its recommendation in order to get the optimal routing.

References

[1] Edward A. Lee. CPS Foundations. (2010)

[2] Eric D. Simmon, Kyoung-sook Kim, Eswaran Subrahmanian, Ryong Lee, Frederic J. de Vaulx, Yohei Murakami, Koji Zettsu, Ram D. Sriram. A Vision of Cyber-Physical Cloud Computing for Smart Networked Systems. (2013)

[3] Ron Banner and Ariel Orda. Multipath Routing Algorithms for Congestion Minimization (2007)

[4] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. Network Flows: Theory, Algorithms, and Applications. (1993)

[5] Kanamori, R., Takahashi, J. and Ito, T. Evaluation of Anticipatory Stigmergy Strategies for Traffic Management. Proceedings of IEEE Vehicular Networking Conference (VNC), pp.33-39 (2012)