Proceedings of the 2019 Winter Simulation Conference
N. Mustafee, K.-H.G. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y.-J. Son, eds.

JAM TAIL ESTIMATION USING VEHICLE AND ROAD AGENTS

Hideyuki Mizuta

IBM Research
19-21 Nihonbashi Hakozaki-cho, Chuo-ku
Tokyo 103-8510, JAPAN

ABSTRACT

Today, an increasing number of vehicles use IoT devices to communicate with a control center to obtain such traffic information as road congestion conditions and the current shortest route. We analyze the enormous amount of data obtained from these vehicles and detect jam tails even if the percentage of vehicles with IoT devices is small. For effective performance and improved accuracy when analyzing an enormous amount of data for a wide road area, we use a multi-agent system to collect and analyze the IoT data, which is stored in memory with a hierarchical structure organized by vehicle agents and road agents. This structure enables time series data to be analyzed from the viewpoint of each vehicle and to be aggregated for jam analysis from the viewpoint of each road. Furthermore, we use a large-scale traffic simulator to evaluate the behavior of this IoT agent system.

1 INTRODUCTION

Today, more than half of the world population lives in urban areas, and the level of urbanization is predicted to rise to 66% in 2050 (United Nations, 2015). One of the key issues for city planners is heavy traffic congestion from the viewpoints of both the environment and economics. By utilizing IoT devices installed in vehicles, we believe that it is possible to decrease the time loss caused by traffic jams.

Recently, an increasing number of vehicles have been using IoT devices to communicate with a control center to obtain traffic information on, for example, road congestion and the current fastest route. We developed a semi-real-time system for analyzing the enormous amount of data obtained from these vehicles and detecting traffic jam tails and their speeds even if the percentage of vehicles with IoT devices is still not sufficient (e.g., 5% or 10%). When analyzing an enormous amount of data for roads over a wide area, for effective performance and improved accuracy, we use a multi-agent system to collect and analyze IoT data, which is stored in memory with a hierarchical structure organized by vehicle agents and road agents.

This structure enables time series data to be analyzed for each vehicle and be aggregated for jam analysis for each road. We also developed a novel algorithm for use with this system to estimate the position of a traffic jam tail.

Furthermore, we use a large-scale traffic simulator to evaluate the behavior of this IoT agent system. The simulated vehicles provide detailed probe-car data for various areas. In this paper, we simulate traffic near the Hakone area of Japan where roads with long traffic jams are often observed. We utilize the simulator as a digital twin of real traffic where each simulated vehicle behaves in the same way as an actual vehicle in the real world, and we can calibrate the IoT analysis algorithm and parameters through the enormous amount of data.

Although there are many services that show road information on the web, they often use only the mean travel time of the road or vehicle density to detect the jam status. In comparison, we analyze a lot of microscopic data obtained from IoT devices installed in vehicles and consider the case in which the percentage of vehicles with IoT devices is not sufficient.

978-1-7281-3283-9/19/$31.00 ©2019 IEEE

2 RELATED WORKS

Nowadays, many services provide traffic conditions, including traffic-jam information, obtained with various sensors. In the map services by Google (Machay 2013), radar-like sensors are utilized to detect traffic congestion across wide areas.

TomTom provides a jam alert service based on the GPS data obtained from floating car data (Clements and Cohn 2016). It aggregates the individual traces of floating cars and detects the region of a jam where vehicle speeds are slower than a threshold. In addition, it considers the influence of severe weather conditions that cause changes in vehicle speeds. Since it uses only the simple analysis of floating car speed to detect jams without estimation, the positions of jam tails become inaccurate when the percentage of installed devices is small.

For developing countries where there is an insufficient number of sensors equipped on the road, in two papers (Katsuki et al. 2017; Idé et al. 2017), techniques were developed for estimating traffic congestion with low-quality cameras. The techniques can estimate the traffic volume of an entire city with a relatively small number of cameras that can be utilized to manage traffic signals (Maeda et al. 2014), but it is difficult to estimate the position of jam tails precisely with fixed cameras.

As classified in a review (Dubey and Borkar 2015), various sensor technologies are applied for research on detecting traffic jams. In this paper, we focus on the GPS data obtained from interconnected vehicles.

Wang et al. (2013) developed an interactive system for visual traffic analysis using GPS trajectories. They compute the traffic speed for each road segment and a jam propagation graph in time and space for a high-level description of a traffic jam. High-level patterns of the state of traffic are also analyzed in Yoon et al. (2007) on the basis of GPS location data. Unique traffic patterns are characterized for each road segment, and unusual traffic states are identified on a segment-by-segment basis. These visual and statistical analyses of traffic can be used to investigate the high-level connections and evolution of traffic states. However, these systems are difficult to apply to our problem for detecting traffic jams in semi-real time with a small sample of devices.

In a series of papers (Schönhof et al. 2007; Kesting and Treiber 2010), the detection of traffic statuses and jams done using inter-vehicle communication is investigated. Probe-car data is used to detect jams on the basis of the speed of vehicles even if the percentage of floating cars is small. The method used in these papers for detecting a jam front from the sudden drop in vehicle speed is similar to our method, but their main objective is to inform the other vehicles of the jam status by using inter-vehicle communication, in which case, the small percentage of vehicles with IoT devices affects the effective information-transfer distance.

From the viewpoint of physical mechanisms, Sugiyama et al. (2008) investigated a traffic jam with a microscopic vehicle. They showed the movement of a jam cluster on a road, but they considered a much smaller situation with only one circular road.

GPS-based techniques can provide a large amount of real-time data. Gupta et al. (2013) proposed a framework for detecting traffic congestion with versatile GPS data coming from various kinds of devices like mobile phones, tablets, and vehicles.

Regarding systems for managing GPS data from IoT devices on connected vehicles, Abe et al. (2017) and Abe et al. (2018) introduced an agent system and rule-based interface for efficient transaction. In this paper, we propose an application for estimating jam tails that utilizes their agent management system for IoT devices.

3 IoT AGENT SYSTEM

We utilize an agent system to collect and analyze vehicle IoT information. Since the amount of this information is so large and the information needs to be processed in semi-real time, a distributed streaming agent system that stores the information in memory and communicates with streaming messages is suitable.

Our agent system was developed with Agent System for IBM IoT Connected Vehicle Insights.

Agents in this system are considered as separate entities that can manage their own sub-entities. Agent managers in the system are executed as servers on different Java VMs and application users can deploy their agents in these agent servers. The interaction between the application and agents is performed by sending and replying to messages via Java communication API or Apache Kafka. An agent handles messages that are sent to it as individual messages or broadcast messages with the corresponding message handler and replies to them.

This agent system is designed for general purpose use including using various IoT applications. To apply this system to IoT automotive analysis, we define vehicle agents and road agents, which correspond to each vehicle with an IoT device (GPS, vehicle sensors, and communication module) and the road.

The probe-car data from IoT devices installed in vehicles are stored in memory with a hierarchical structure organized by vehicle agents and road agents. This structure enables time series data to be analyzed for each vehicle and be aggregated for jam analysis for each road. To reduce system resources, we generate vehicle and road agents when the application receives corresponding probe-car data.

A simple data transfer process is shown in Figure 1. An executor created by an IoT application can move into the agent space and send messages to agents, and the agents reply to these messages so that the IoT application can receive the traffic jam information.

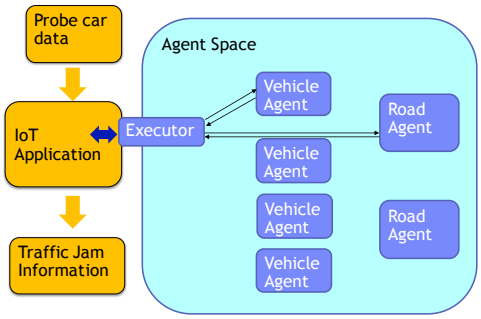

Figure 1: Data and message flow in IoT agent system.

When the application receives probe-car data, it sends an executor with probe data to the agent world. The executor in the agent world generates the corresponding vehicle agent and road agent if they do not exist and sends a message to the vehicle agent for the first step of analysis. If the speed of the vehicle is low (i.e., it is not cruising freely), it sends a message containing the vehicle data with the result of the first step to the corresponding road agent. After analyzing the road agent with the aggregated vehicle data on the road at that timestep, the executor and the application receive a reply with the jam analysis result from the road agent.

The vehicle and road agents in this system can be located at distributed computation nodes and executed in parallel for semi-real-time transactions.

4 JAM TAIL ESTIMATION

Now, we describe the algorithm for estimating jam tails from probe-car data. The probe-car data consists of the time, vehicle ID, road ID, speed, length from the start point of the road (position), speed limit of the road, longitude, and latitude. Before this analysis, the data is processed with a map matching process, and the road and position on the road are given.

First, we analyze the time series data of vehicle speed with the corresponding vehicle agent. The vehicle agent stores a time series of vehicle speed (v_t) and the difference in vehicle speed ($\Delta v_t = v_t - v_{t-1}$). Generally, vehicle speed changes frequently over time, and we need to smooth the variance. In this paper,

we use the decaying average for smoothing. With the coefficient $\alpha = 0.1$, the average speed is updated as $\tilde{v}_t = (1-\alpha)\tilde{v}_{t-1} + \alpha v_t$.

First, we simply classify the jam status of a vehicle as FREE, SLOW, JAM, or STOP in accordance with the average speed (Figure 2).

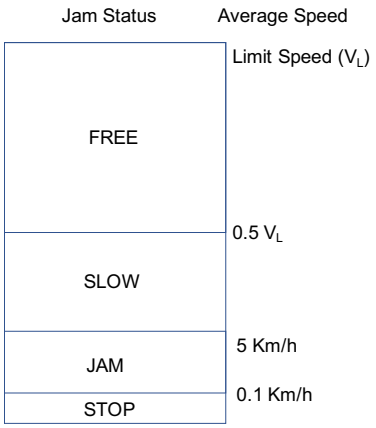

Figure 2: Classification of jam status and vehicle speed.

We also consider the difference in the average moving speed as the speed trend $T = (v_t - v_{t-w_1})/w_1$, where $w_1 = 9$ is the time window for the average moving speed. When a vehicle enters into a jam group, the speed significantly decreases. Hence, we utilize the speed trend to detect the exact time and position of the jam tail.

Furthermore, we aggressively utilize the fluctuation in vehicle speed (wave) that occurs due to interaction with other vehicles.

For wave detection, we use the wave trend $T_W = \sum_{t-w_2+1}^{t} \Delta v_t / w_2$ and wave variance $V_W = \sum_{t-w_2+1}^{t} |\Delta v_t| / w_2$ defined with the moving average with time window $w_2 = 30$ of the difference in speeds. If the jam status is JAM or STOP, the wave trend is small ($|T_W| \leq \beta$), wave variance is in a fixed range ($\gamma_L \leq V_W \leq \gamma_H$), and the status of the vehicle is classified as wave. Figure 3 shows an example of v_t, $\tilde{v}_t$, T_W, and V_W for one vehicle. In this paper, the parameters are $\beta = 1.0, \gamma_L = 0.5, and \gamma_H = 7.0$

To avoid a sudden change in wave status and modify the status in accordance with the jam status, we consider the wave rate R_W ($0 \leq R_W \leq 2$). If the vehicle status is classified as wave and the current jam status is JAM or STOP, R_W is increased ($R_W = R_W + r_1$). Otherwise, R_W is decreased, $R_W = R_W + r_j$, in accordance with the jam status j (Figure 4). Finally, the jam status is set to WAVEJAM if $R_W \geq 1$.

The procedures for the vehicle agent are shown in Figure 5.

Vehicle agents store the information on the jam status with the probe-car data to send it to the corresponding road agents and reply to the application. Vehicle agents also store the time they enter the current road (entry time) for further analysis.

After classifying the jam status of vehicle agents, we consider the analysis of road agents. The vehicle data is sent to corresponding road agents only when the jam status is not FREE.

Now, we describe the analysis for road agents. The main task of the road agent is to estimate traffic jam tails on the basis of the aggregated information on the vehicle agents on the road. If a road agent cannot find a jam tail, it means that there is no jam on the road at that particular moment. The procedure for the road agent is summarized as follows.

1. Sort the vehicle data from the end of the road.
2. Find the group of vehicles with jam speed.
3. Find the vehicles with the wave status.
4. Find the vehicles with a large speed decrease.

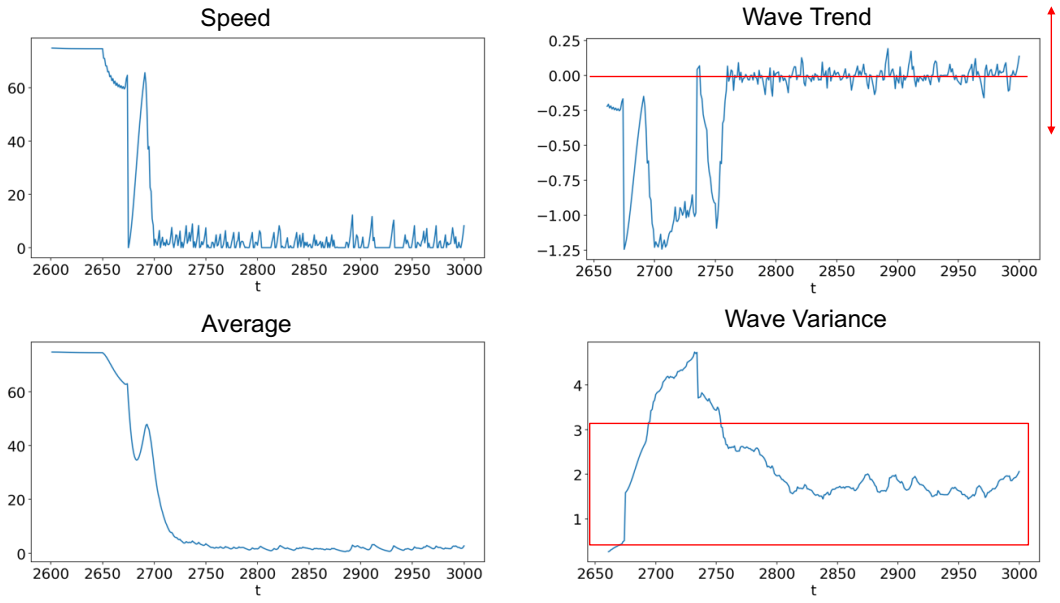

Figure 3: Time series of speed and average speed, wave trend and wave variance.

5. Adjust the position of the previous jam tail.

The road agent stores the received vehicle data with timestamp t and executes analysis when it receives a message with timestamp $t+1$ to analyze all (non-FREE) vehicles on the road at time t. The road agent also has the previous position of the jam tail (P_{t-1}) and corresponding vehicle ID (pvid). If pvid is found in the stored vehicle data at time t, the corresponding vehicle speed (v_p) is recorded for the analysis.

When N vehicles exist at time t, we first calculate the mean interval time. Each vehicle records its entry time t_E for the current road. The mean interval time I is given by $I = (\max(t_E) - \min(t_E))/(N-1)$.

Before further analysis, the vehicle data is sorted by position p_i in descending order so that the front vehicle is analyzed first ($p_1 \geq p_2 \geq \ldots \geq p_N$). We can calculate the gap g_i between the vehicles after the sorting as $g_i = p_i - p_{i+1}$. The maximum value $G = \max(g_i)$ of the gaps will be used to adjust the tail position.

If the number of vehicles with a JAM or STOP jam status is larger than threshold N_J, we set the jam status of the road as JAM and record the smallest position (position nearest the origin of the road) as the temporary jam tail. We also consider the status of a road to be JAM if the number of vehicles with the WAVEJAM jam status is larger than the smaller threshold $N_W \leq N_J$, and we record the temporary jam-tail position since the wave status indicates the existence of vehicles in a jam even if the number of vehicles with an installed IoT device is small on the road. For sampled vehicle data, we use $N_J = 2$ and $N_W = 1$.

The position of a jam tail moves as the tail vehicle moves and is updated if a new vehicle is added to the road. However, there may be a vehicle that cannot be observed because it does not have an IoT device. Hence, we need to estimate the true position of the jam tail. If a vehicle with a large decrease in speed $T \leq -\tau$ ($\tau > 0$) exists, we can consider the position of this vehicle as the true position of the jam tail and store the position in the road agent entity (e.g. $\tau = 2$).

When such a true tail vehicle does not exist and the road agent has stored the previous tail position and previous vehicle ID, we adjust the position on the basis of the stored position P_{t-1} with $P_t = P_{t-1} + v_p - G/I$, where v_p is the vehicle speed of the previous tail vehicle, G is the maximum value of the gap between vehicles, and I is the average of the interval times. Otherwise, we use the temporary jam tail as jam tail P_t and store it and the corresponding vehicle ID.

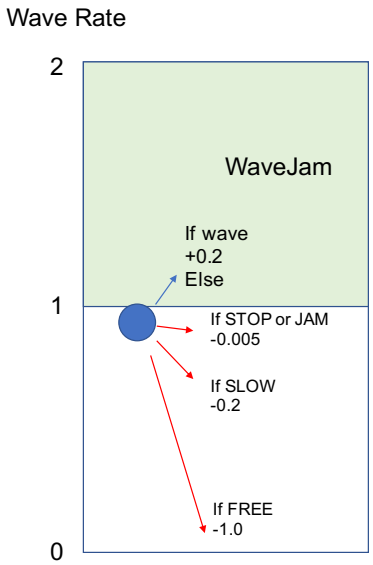

Figure 4: Change in wave rate in accordance with wave status and jam status

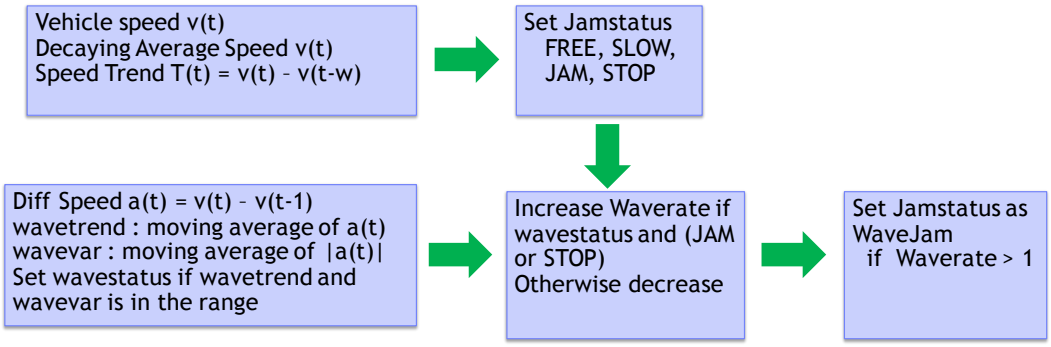

Figure 5: Process for setting jam status for vehicle agents

Either a calculated jam position P_t or a flag indicating that we cannot find a jam on the road are recorded.

Finally, the calculated jam position is sent back to the application through the vehicle agent and executor. The jam information for each road and time are aggregated in the application and used for macro analysis or visualization.

5 LARGE-SCALE AGENT-BASED TRAFFIC SIMULATION

For this paper, we evaluated the jam detection system with probe-car data generated by an agent-based traffic simulator. Agent-based simulation is a powerful tool for understanding complicated dynamic systems, such as those of an entire city including many people, and for reproducing such activities. We utilized IBM Mega Traffic Simulator (Osogami et al. 2013) to generate microscopic vehicle data.

This agent-based traffic simulator is a standalone application and considers each microscopic vehicle as an agent that travels through a given road network with crosspoints (node) and roads (links). Each agent is assigned an origin, a destination, and a departure time for a trip in accordance with an origin-destination (OD) table of the traffic demand.

The simulator creates an agent at an origin at a departure time. The agent chooses a route from the origin to the destination in accordance with the shortest travel time estimated at the departure time and travels along that route. In this simulator, heterogeneous agents (drivers) select a route and change their car speed and lane on the basis of a car following model (Gipps 1981) and the Integrated Lane-Changing Model (Toledo et al. 2003), which represent the dynamic interaction with surrounding cars. At each timestep (typically, 1 sec), the microscopic car behavior on the roads is controlled by connected crosspoints, each of which is assigned an individual thread from a thread pool to effectively simulate fine-grained car movement on and across roads even in a distributed HPC environment. Though almost all behaviors of a vehicle are deterministic, the lane changing behavior is probabilistic. The simulator tracks the location of each agent and records the information on the vehicles (position and speed), roads (average speed, number of vehicles, CO2 emissions on the road), and trips (travel time and total CO2 emissions of each vehicle) into log files that are used for analysis and visualization. For the test data, we utilized the log data of vehicles consisting of time, vehicle ID, road ID, speed, length from the start point of the road (position), longitude, and latitude. We also used road network data to obtain the speed-limit speed of each road.

For the simulation, we extracted road network data (node and link) from OpenStreetMap . The extracted area includes Hakone, Japan, where heavy traffic jams are often observed because of the large amount of traffic between the west and east areas of Japan (Figure 6). The map image is based upon OpenStreetMap data and licensed under CC-BY-SA 2.0 or ODbL.

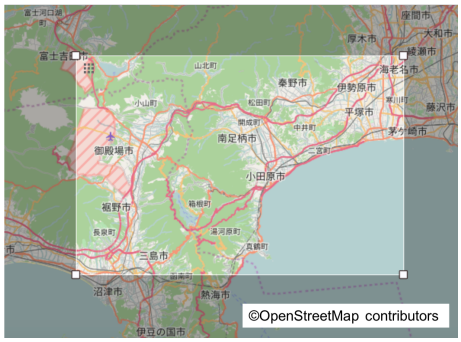

Figure 6: Map area near Hakone for road network extraction.

For this area, we assumed a simple OD table that divides the area into four sub-areas (Figure 7). With this traffic demand, we generated a trip definition with a randomly selected origin and destination crosspoints.

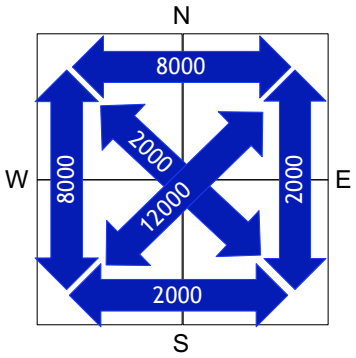

Figure 7: OD table representing traffic demand for 1 hour.

With these input data, we performed the simulation for 1 hour (3600 timesteps). From the vehicle log data, we extracted the last 20 minutes of data to omit the initial status with a small number of vehicles.

Figure 8 shows the position and speed of one vehicle (Vehicle View), and Figure 9 shows the position of all vehicles on one road for 20 minutes (Road View).

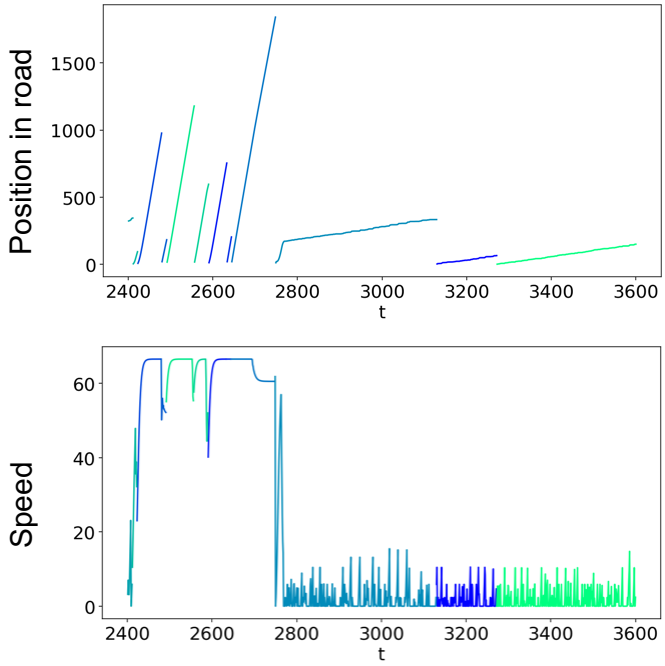

Figure 8: Position and speed of one vehicle.

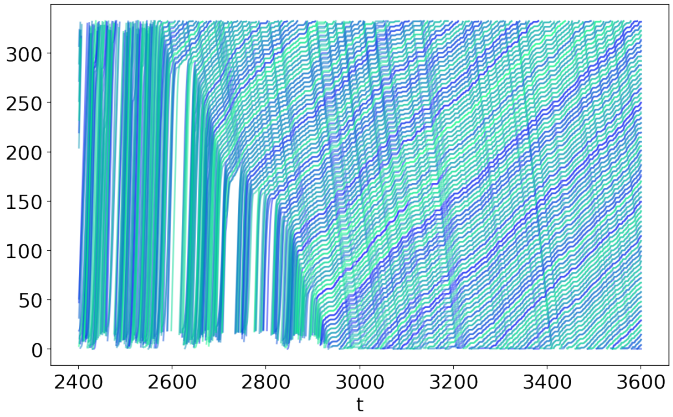

Figure 9: Position of vehicles on one road.

In the jam-estimation step, we utilized both views. We can see the wave and sudden change in speed in the road view that shows the jam region.

6 EVALUATION OF JAM TAIL ESTIMATION

In this section, we evaluate our procedure for detecting jam statuses and estimating jam tails with the traffic simulation data.

Our application read the log data of the probe-car data generated by the traffic simulator and output the jam-tail position on the road where a traffic jam was detected at each timestep (1 second). To consider a case in which the number of vehicles with an IoT device installed is limited, we applied the method with sampled data for 20%, 10%, and 5% of vehicles having IoT devices.

We first evaluated how many roads were detected as jam roads by comparing the road IDs of the output data at each time between the full vehicle input data, which is data assuming that all cars have IoT devices, and sampled vehicle input data. The number of jam roads was relatively small compared with the number of roads without traffic jams. We focused on roads where traffic jams were correctly detected (i.e. the jam-tail position was reported). Then, we compared the position of the jam tail when both types of input data output the same road ID. To evaluate our method with wave detection, decaying average speed, and jam-tail estimation, we compared the results with a simple method (Speed) that only considers the vehicle speed as in the literature with GPS data and another method (Decay) for inter-vehicle communications (Schönhof et al. 2007; Kesting and Treiber 2010) that uses the decaying average speed. These simplified methods are typical traditional methods and only use the threshold of speed to detect traffic jams.

The output of our system is a list of jam-tail positions on each road with a traffic jam at each time. In this evaluation, we used the simulation data for 1200 seconds (T). Hence, if there are traffic jams on all roads, the number of jam-tail positions in the output will be $T \times N_R$ (the number of roads). By comparing the existence of jam information at each time at each road between the output with full-vehicle data and the output with sampled data, we can calculate the number of correct detections (True Positive, TP), the number of missed detections (False Negative, FN), and the number of false detections (False Positive, FP). The number of roads correctly classified as having a non-jam status (True Negative, TN) depends on the total number of pieces of road status data ($N_D = TN_R$). In this evaluation, we did not use all roads on the map but rather the roads where at least one vehicle existed at each time (the maximum number of possible detectable jams). For intuitive description, we use the numbers detected per second.

Table 1 shows a comparison of the average jam detection (TP, FN, FP, TN) with different sample percentages and methods in consideration of the analysis of full-vehicle data, which represent the true results. For the evaluation, we generated eight pieces of simulation log data with different random seeds. Though the Decay method showed low false-positive values, the proposed method showed low false-negative values that are more important for safety.

Table 1: True and false jam detection per sec with 5%, 10%, and 20% sample input data.

Sample (%)	Method	TP	FN	FP	TN
20	Proposed	267.97	7.91	42.59	4400.15
20	Decay	253.26	22.62	22.23	4420.52
20	Speed	250.23	25.65	49.45	4393.29
10	Proposed	247.31	28.57	25.72	4417.03
10	Decay	208.38	67.5	8.68	4434.06
10	Speed	203.96	71.92	22.23	4420.51
5	Proposed	206.54	69.34	14.19	4428.54
5	Decay	146.76	129.12	3.21	4439.54
5	Speed	142.27	133.61	8.92	4433.83

For Table 1, we computed the accuracy of each method with $(TP+TN)/(TP+TN+FP+FN)$. Similarly, we calculated the precision, which indicates the ratio of correctly detected traffic jams, with $TP/(TP+FP)$ and the recall rate, which indicates the rate of detected traffic jams, with $TP/(TP+FN)$. In addition, we calculated the F1 score as the harmonic average of precision and recall.

The results with our method outperformed the simplified method in terms of recall, F1 score, and error in jam-tail position (Table 2). In particular, the recall rate significantly improved with our method. With a

Table 2: Evaluation of jam detection rate and error in jam-tail position with 5%, 10% and 20% sample input data.

Sample (%)	Method	Accuracy	Precision	Recall	F1	Position Error (m)
20	Proposed	98.93	86.32	97.15	45.71	13.06
20	Decay	99.06	91.95	91.84	45.95	16.66
20	Speed	98.41	83.51	90.75	43.49	20.66
10	Proposed	98.85	90.61	89.68	45.07	20.3
10	Decay	98.36	96.01	75.62	42.29	28.23
10	Speed	98.03	90.17	74.02	40.64	30.76
5	Proposed	98.25	93.59	74.93	41.61	29.89
5	Decay	97.2	97.87	53.29	34.49	45.29
5	Speed	96.98	94.11	51.67	33.34	46.68

sample rate of only 5% (vehicles with IoT devices), our method still detected 75% of road jams at each moment. When there was only a small number of vehicles or no vehicles on the road, it is difficult or impossible to detect jams on the road. This missing information both in time and road decreased the recall rate. Even in such a case, our method detected traffic jams when there was a small number of vehicles with the wave status, which was caused by the presence of surrounding vehicles that may not have been installed with IoT devices.

The reason that the precision with small samples and the Decay method was larger than that of larger samples and our method was simply because the number of false positives was very small, which means more jams were detected than for full data. In other words, recall and precision values are in a trade-off relationship. We consider the recall rate to be more important for correctly planning a route and avoiding the risk of colliding with a preceding vehicle in a jam.

Figure 10 and 11 show a box chart of recall, precision, accuracy, and position error with each method and sample percentage.

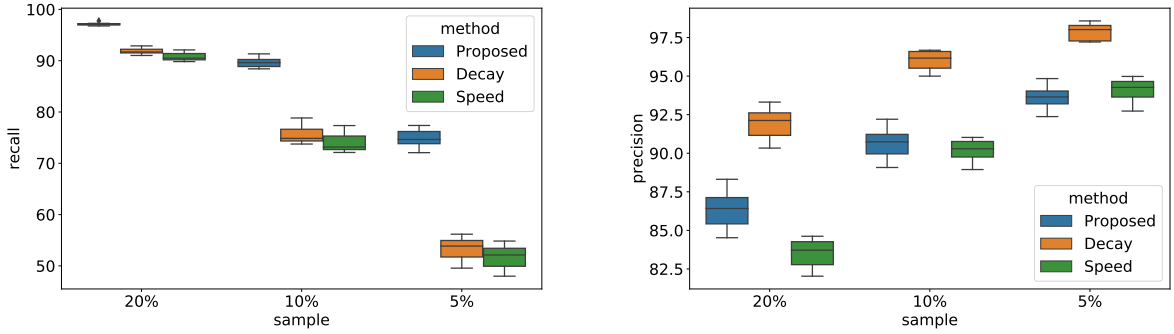

Figure 10: Comparison of recall and precision using proposed method, decaying average speed, and simple speed method with sample percentages.

Figure 12 shows a detailed estimation of jam-tail position by showing the trajectories of vehicles on a road. In the left region (earlier time), vehicles ran at high speed as is indicated by the high slope of the lines. After a while (right region), vehicles entered into a traffic jam at low speed. The tail of the traffic jam is observed in the border region.

Since the sampled input data had only small number of vehicles that appeared with long intervals, the estimated jam tail tended to depart from the true jam tail shown with the black line. However, the adjusted jam tail with our method (blue line) shows a good approximation of the true position.

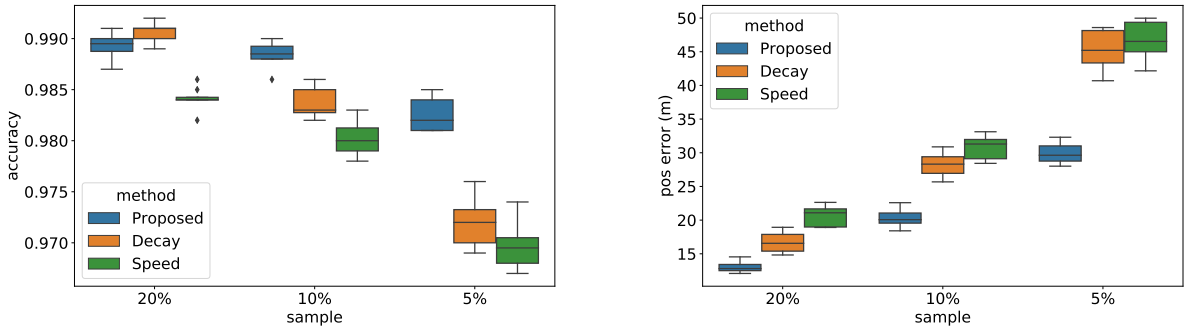

Figure 11: Comparison of accuracy and error in jam-tail position using proposed method, decaying average speed, and simple speed method with sample percentages.

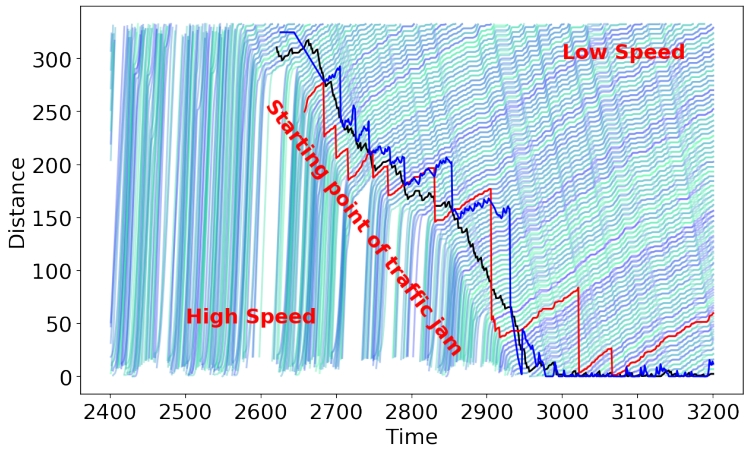

Figure 12: Comparison of jam-tail estimation among full-vehicle data (black) and sample input data (5%) with our method (blue) and traditional methods without adjustment (red).

7 CONCLUDING REMARKS

In this paper, we introduced our procedure for detecting and estimating jam tails.

For effective performance, we used a multi-agent system to collect and analyze IoT data, which is stored in memory with a hierarchical structure organized by vehicle agents and road agents. This structure enables time series data to be analyzed for each vehicle and be aggregated for jam analysis for each road. In particular, we analyzed the wave-like behavior of vehicle speed change to detect a jam even when only 5% of the vehicles had IoT devices installed.

We also utilized the probe-car data generated by an agent-based traffic simulator. Such utilization of a simulator for the evaluation of a new algorithm will be valuable in many research and development areas.

The results show that our method outperformed the traditional methods without wave detection and jam-tail estimation. In particular, the recall rate significantly improved with our method. The estimation of the jam-tail position also improved with our method through the adjustment of the positions with gaps and intervals even though very few vehicles appear on the road in a small-sample case.

For future work, we will consider applying the proposed system to real traffic data. We need to modify the parameters for such data in various countries and regions. Though the parameters in this paper were determined by hand after experiments and visualization, the utilization of machine learning methods with

an enormous amount of data from traffic simulation and real vehicles will help us to automatically adjust such parameters.

REFERENCES

Abe, M., G. Yamamoto, and S. Furuichi. 2018. "(WIP) IoT Context Descriptor: Situation Detection and Action Invocation Model for Real-time High-Volume Transactions". In *2018 IEEE International Congress on Internet of Things*, 130–133. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Abe, M., G. Yamamoto, and T. Miyahira. 2017. "Rule-based Situation Inference for Connected Vehicles". In *2017 IEEE International Congress on Internet of Things*, 159–161. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Clements, M. and N. Cohn. 2016. "Real-time Safety Alerts for Severe Weather and Jam Tails". In *European Transport Conference 2016 Association for European Transport*. October 5th–7th, Barcelona, Spain.

Dubey, P. P. and P. Borkar. 2015. "Review on Techniques for Traffic Jam Detection and Congestion Avoidance". In *2015 2nd International Conference on Electronics and Communication Systems*, 434–440. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Gipps, P. G. 1981. "A Behavioural Car-Following Model for Computer Simulation". *Transportation Research Part B: Methodological* 15(2):105–111.

Gupta, A., S. Choudhary, and S. Paul. 2013. "DTC: A Framework to Detect Traffic Congestion by Mining Versatile GPS Data". In *2013 1st International Conference on Emerging Trends and Applications in Computer Science*, 97–103. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

IBM. 2019. "Agent System for IBM IoT Connected Vehicle Insights". https://www.ibm.com/support/knowledgecenter/en/SSNQ4V_bas/iot-automotive/managing/agent/agent_intro.html, accessed 28th June.

Idé, T., T. Katsuki, T. Morimura, and R. Morris. 2017. "City-wide Traffic Flow Estimation from a Limited Number of Low-Quality Cameras". *IEEE Transactions on Intelligent Transportation Systems* 18(4):950–959.

Katsuki, T., T. Morimura, and M. Inoue. 2017. "Traffic Velocity Estimation from Vehicle Count Sequences". *IEEE Transactions on Intelligent Transportation Systems* 18(7):1700–1712.

Kesting, A. and M. Treiber. 2010. "Online Traffic State Estimation Based on Floating Car Data". *arXiv Preprint arXiv:1012.4567*.

Machay, J. 2013. "How Does Google Detect Traffic Congestion?". https://smallbusiness.chron.com/google-detect-traffic-congestion-49523.html.

Maeda, K., T. Morimura, T. Katsuki, and M. Teraguchi. 2014. "Frugal Signal Control Using Low Resolution Web-Camera and Traffic Flow Estimation". In *Proceedings of the 2014 Winter Simulation Conference*, edited by A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, 2082–2091. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

OpenStreetMap. 2019. http://openstreetmap.org/, accessed 28th June.

Osogami, T., T. Imamichi, H. Mizuta, T. Suzumura, and T. Ide. 2013. "Toward Simulating Entire Cities with Behavioral Models of Traffic". *IBM Journal of Research and Development* 57(5):6–1.

Schönhof, M., M. Treiber, A. Kesting, and D. Helbing. 2007. "Autonomous Detection and Anticipation of Jam Fronts from Messages Propagated by Intervehicle Communication". *Transportation Research Record* 1999(1):3–12.

Sugiyama, Y., M. Fukui, M. Kikuchi, K. Hasebe, A. Nakayama, K. Nishinari, S.-i. Tadaki, and S. Yukawa. 2008. "Traffic Jams without Bottlenecks – Experimental Evidence for the Physical Mechanism of the Formation of a Jam". *New Journal of Physics* 10(3):033001.

Toledo, T., H. Koutsopoulos, and M. Ben-Akiva. 2003. "Modeling Integrated Lane-Changing Behavior". *Transportation Research Record: Journal of the Transportation Research Board* (1857):30–38.

Wang, Z., M. Lu, X. Yuan, J. Zhang, and H. Van De Wetering. 2013. "Visual Traffic Jam Analysis Based on Trajectory Data". *IEEE Transactions on Visualization and Computer Graphics* 19(12):2159–2168.

Yoon, J., B. Noble, and M. Liu. 2007. "Surface Street Traffic Estimation". In *Proceedings of the 5th International Conference on Mobile Systems, Applications and Services*, 220–232. New York: Association for Computing Machinery, Inc.

AUTHOR BIOGRAPHY

HIDEYUKI MIZUTA is a Research Staff Member of IBM Research – Tokyo. He received B.S., M.S., and Ph.D. degrees in Physics from the University of Tokyo. He is a member of IPSJ and ACM SIGSIM. His research interests include dynamic economic-social systems with heterogeneous agents, SSME (Services Science, Management and Engineering) and smarter cities. His email address is e28193@jp.ibm.com.