Fast Railway Delay Evaluation Method Based on Discrete Distribution Propagation

Tomoshi Otsuki[1,a)] Hideo Sakamoto[1] Hideki Kubo[2]

Abstract: Accurate delay evaluation of timetables is crucial for railway companies. There have been some conventional methods utilizing continuous random variables directly. These methods, however, suffer combinatorial expansion problems, which require complex pruning techniques. In this paper, discretizing the delay distribution on railway networks, we present a method calculating propagated delay distributions on each event analytically under the assumption of propagated delays' independence. We also show the complexity of the proposed method is $O(M^2N)$ in the general case and can be reduced to $O(MN)$ especially for the special cases where the source distribution is the negative binominal distribution, where M denotes the number of quantization levels in discretization and N denotes the total number of events. Finally, computational experiments on test scenarios with $M \sim 700$ show that the proposed method provides almost same results as and in over 500 times faster computation time than Monte Carlo simulations.

Keywords: Railway scheduling, Train delays, Delay propagation, Robustness

1. Introduction

Accurate delay evaluation of timetables is crucial for railway companies, since train delays cause dissatisfaction of passengers, reduction in revenue, and increase in the penalty payment for compensation when the delay exceeds some threshold time.

Thus, they usually give some buffer to timetables in preparation for the delay. If we increase buffer time, the frequency of delays can be reduced. However, too much buffer time increases the service time, causing the dissatisfaction of passengers. Thus, a trade-off exists between punctuality and service levels of train operations.

To evaluate timetables, we introduce the random variables representing delays in each station of timetables.

Railway timetables consist of multiple sequences corresponding to respective trains each of which consists of multiple scheduled events such as departures, arrivals, and passings. Train delays occur between two events. For example, the delay while running occurs between the departure event at a station and the arrival event at the following station.

Train delays are categorized into two types. First, we define a *source delay* as a primary delay on each train typically due to technical failures or unexpected passenger's behaviors. Then a *propagated delay*, on the other hand, is a secondary delay caused by preceding events of train itself or different trains when accumulating delays exceed some buffer time.

In this paper, we propose the way of calculating the propagated delays on each event where source delays are given.

The remainder of this paper is organized as follows. Section 2 provides related work. Then Section 3 gives a problem setting and stochastic modelling of propagated delays. Section 4 gives an algorithm and its complexity. Section 5 reports comparison results between the proposed approach and conventional Monte Carlo simulation-based approach. Finally, Section 6 provides a conclusion.

2. Related Work

To evaluate propagated delays, there have been mainly three approaches: deterministic approaches, Monte Carlo simulation-based approaches, and stochastic approaches.

First of all, deterministic approaches for timetable simulation have been well studied for a long time. PERT (Program Evaluation and Review Technique), which can find out critical paths of delays, is one of the methods which can be utilized in cases such as diagram simulation [3], rescheduling [4], and shunting scheduling [5]. However, since PERT deals with deterministic variables, it cannot calculate the distribution of delays.

Then Monte Carlo simulation-based approaches (MC) can evaluate propagated delays as well. For example, Ushida *et al.* [6] reports a method of evaluating robustness of timetables on a railway company. Then Tatsui *et al.* [7] proposes a method simulating timetables based on predicted number of passengers by neural network and Nakamura *et al.* [8] and Takeuchi *et al.* [9] evaluate the robustness of railway

1 Toshiba Corporation
1, Komukai-Toshiba-cho, Saiwai-ku, Kawasaki 212–8582, Japan
2 Toshiba Digital Solutions Corporation
72-34, Horikawa-Cho, Saiwai-ku, Kawasaki 212–8585, Japan
a) tomoshi1.otsuki@toshiba.co.jp

timetable based on statistics of passengers. However, though MC can evaluate timetables accurately if there are the adequate number of iterations, it requires much time to run many iterations. Moreover, for running detailed simulation, we need to collect many kinds of accurate information and to handle missing or error data, which usually takes a lot of time and effort. On the other hand, our proposed method can be executed only by easily-accessible operation records such as run and dwell time statistics or, more simply, by the average of run and dwell time.

Finally, stochastic approaches directly process cumulative distribution functions (CDF) of underlying random variables. To deal with the CDF of propagated delays on the network similar to ours, they propose some probability distribution classes which have the closeness under required operations. For example, Buker *et al.* [2] and Kirchhoff *et al.* [10] propose the distribution classes that consist of the sum of extended exponential polynomials. However, since the number of terms grows exponentially, they require pruning techniques. On the other hand, Measter *et al.* [1] proposes the phase-type-distribution for propagated delays. However, since the size of matrix grows exponentially, they proposed the method approximating the matrix with an upper triangular one.

3. Problem Setting

3.1 Network Model

Consider a directed network $G(V, A)$ comprising node set V and arc set A to represent delays on the timetable. The node set V represents events on a timetable and the arc set A represents the possibility of delay propagation between events on both side of the arc. For each node $v \in V$, *scheduled time* t_v^0 is set to represent the pre-determined time of the node v on the timetable.

For simplicity, we deal only three node types in this paper as shown in Figure 1 as follows:

- *departure node* representing a departure from one of the stations,
- *arrival node* representing an arrival at one of the stations, and
- *passing node* representing a passing through one of the stations.

Then a directed arc $a_{ij} \in A$ that connects nodes i and j, means the possibility of a delay of the node i can be propagated to the node j. Note that the direction of arcs is determined to satisfy $t_i^0 < t_j^0$.

Though source delays are caused by several factors including boarding passengers, turnarounds, platform tracks, or junctions, we consider only three typical arc types in this paper for simplicity:

- *run arc* representing a running phase of a train between a departure or passing node and the following passing or arrival node.
- *dwell arc* representing a dwell phase of a train between an arrival node and the following departure node.
- *headway arc* representing a delay propagation possibility between different trains, which connects a departure node of one train and the consecutive arrival node of the following train.

3.2 Modelling Delays on the Network

For a node pair i, j where $a_{i,j}$ is defined, we also define *minimum arc time* that represents the minimum required time of the arc $a_{i,j}$ and *buffer time* that represents the marginal time set to $a_{i,j}$.

Then the interval of scheduled times between t_i^0 and t_j^0 is decomposed into minimum arc time $h_{i,j}(> 0)$ and the residual buffer time $b_{i,j}(\geq 0)$ as follows:

$$t_j^0 = t_i^0 + h_{i,j} + b_{i,j}. \tag{1}$$

Then consider actual operations on the graph. Let $d_{i,j}$ be a random variable representing delay time on an arc $a_{i,j}$, which follows a distribution $P_{i,j}$, and let t_j be an *operation time* at which event j actually occurs.

We assume source delays are generated only on run and dwell arcs and no source delays are generated on headway arcs. Thus, on a headway arc, we define $P_{i,j}$ as taking all the probability mass on the delay-zero-point for simplicity.

Moreover, we assume:

(1) The delay of a node j is affected by the delay from multiple nodes $i(\in \mathbf{prev}(j))$, where $\mathbf{prev}(j)$ is a set of nodes immediately before node j. Then the delay of j takes maximum of propagated delays from multiple previous nodes.

(2) On any node j, operation time t_j is never earlier than scheduled time t_j^0, since departing earlier than scheduled time is banned in many railway companies.

Then the operation time t_j on node j is represented as follows:

$$t_j = \max\{t_j^0, \max_{i \in \mathbf{prev}(j)} \{t_i + h_{i,j} + d_{i,j}\}\}, \quad d_{i,j} \sim P_{i,j}. \tag{2}$$

Consider a random variable $X_j = t_j - t_j^0$ representing the delay from scheduled time on node j. Then from Equations (1) and (2), we have

$$X_j = [\max_{i \in \mathbf{prev}(j)} \{X_i + d_{i,j} - b_{i,j}\}]^+, \tag{3}$$

where $[x]^+ \equiv \max\{x, 0\}$.

When we evaluate each node in the topological order, we can calculate all the propagated delays recursively by Equation (3).

4. Delay Propagation Algorithm and Its Complexity

4.1 Discretization of Random Variables

In the discussion above, we have regarded propagated delays X as continuous random variables. However, handling continuous random variables analytically suffers combinatorial expansion problems as shown in Section 2.

Thus, we discretize the random variable X, and deal the probability mass function (PMF) in the range from $k = 1$ to

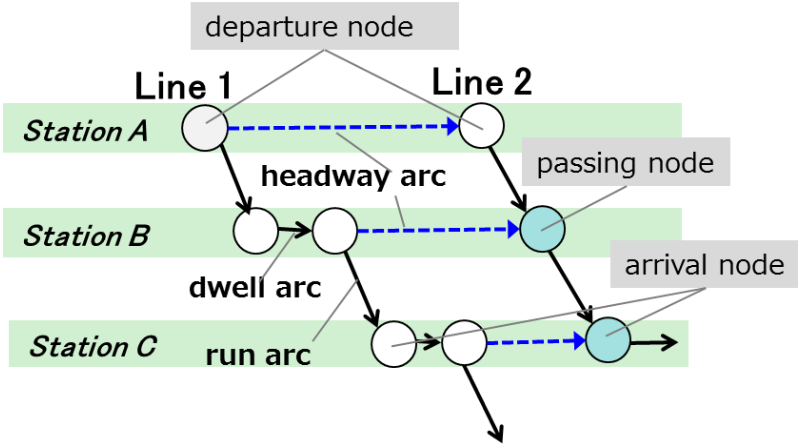

Fig. 1 Directed network representing scheduled events and propagated delays. Gray nodes represent the nodes in which the number of incoming arcs is two.

$k = M$. When we set sufficiently large M, the PMF is considered to approach the continuous probability distribution function (PDF).

Note that when the delay time takes negative values, we shift the domain of the distribution to $k \geq 1$ without loss of generality. In this case, let $k = k_0$ be the delay-zero-point in the original distribution.

4.2 Assumption

In our proposed method, we will make the following two assumptions:

S_Independence All the source delay distributions are independent.

P_Independence All the propagated delay distributions from different routes are independent.

The first assumption means the independence of all $P_{i,j}$s. This is natural assumption since small-scale delays usually occur independently.

Additionally, as in previous studies including [1][2] and [10], we assume the second assumption, which indicates the independence of all the preceding nodes' delays $X_i (i \in \textbf{prev}(j))$ when we evaluate X_j in Equation (3). This assumption can be violated in general since our network is grid-like and all the routes to a node originate from the initial node (e.g. the node at Station A on Train 1 in Figure 1). We will discuss this issue again in Section 5.

Moreover, for simplicity, we denote $\Pr(X = k)$ by $X[k]$.

4.3 Calculation of Propagated Delays in Each Node

The process of calculating X_j in each node j from previous nodes' propagated delays $(\{X_i\}|i \in \textbf{prev}(j))$ can be written in the following Algorithm 1, where $\mathrm{CONV}(X, D)$ is a convolution of X and D, $\mathrm{SHIFT}(Y, b)$ is a shift operation to Y by a constant value b, $\mathrm{GETMAX}_{i \in \textbf{prev}(j)}(\{Z_i\})$ is a maximum operation among Z_is, and $\mathrm{FLOOR}(W, k_0)$ is an operation flooring a W by a constant value k_0.

Algorithm 1 The procedure of getting propagated delays on node j

Require: j: current node id
Require: $D_{i,j}$: discrete distribution of source delays on $a_{i,j}$
Require: $b_{i,j}$: buffer time on $a_{i,j}$
Require: $\textbf{prev}(j)$: previous node set of node j
Ensure: X_j: discrete distribution of propagated delays on j

1: **for all** $i \in \textbf{prev}(j)$ **do**
2: $\quad Y_i \leftarrow \mathrm{CONV}(X_i, D_{i,j})$
3: $\quad Z_i \leftarrow \mathrm{SHIFT}(Y_i, b_{i,j})$
4: **end for**
5: $W_j \leftarrow \mathrm{GETMAX}_{i \in \textbf{prev}(j)}\{Z_i\}$
6: $X_j \leftarrow \mathrm{FLOOR}(W_j, k_0)$

When we consider the fact that distribution of the sum of two independent random variables can be represented as convolution, Algorithm 1 strictly corresponds to $X_j = [\max_{i \in \textbf{prev}(j)}\{X_i + d_{i,j} - b_{i,j}\}]^+$ in Equation (3).

Here, since X and D are independent under **S_Independence** assumption, the result PMF of $Y \leftarrow \mathrm{CONV}(X, D)$ is obtained as follows:

$$Y[k] = \sum_{i=1}^{k} X[i] \cdot D[k-i] \quad (k = 1, 2, \ldots, M), \qquad (4)$$

where $X[k]$ and $D[k]$ are the inputs of the convolution.

Then the result PMF of $Z \leftarrow \mathrm{SHIFT}(Y, b)$ is

$$Z[k] = Y[\max\{1, k-b\}] \quad (k = 1, 2, \ldots, M), \qquad (5)$$

where $Y[k]$ and $b (= 0, 1, 2, \ldots)$ are the inputs of the shift operation.

Then consider the result MPF of $W \leftarrow \mathrm{GETMAX}_{l \in L}\{Z_l\}$, where L be the number of input variables. Let C_l be CDF of Z_l. Then since all the C_ls are independent under **P_Independence** assumption, the probability that all $\{C_l\}_{l=1}^{L}$ are k or less is represented as $\prod_{l=1}^{L} C_l[k]$. Let $W[k]$ represent the probability that the maximum of $\{C_l\}_{l=1}^{L}$ is just k. Then $W[k]$ is obtained by subtracting the probability that all the variables are under k from the probability under $k+1$. Thus, we obtain $W[k]$

as follows:

$$W[k] = \begin{cases} \prod_{l=1}^{L} C_l[1] & (k=1) \\ \prod_{l=1}^{L} C_l[k] - \prod_{l=1}^{L} C_l[k-1] \\ & (k=2,3,\ldots) \end{cases} \quad (6)$$

Thus, the complexity of $\mathrm{GETMAX}_{l \in L}\{Z_l\}$ is $O(LM)$.

Moreover, we have the following result PMF of $X \leftarrow \mathrm{FLOOR}(W, k_0)$:

$$X[k] = \begin{cases} W[k] & \text{if } k > k_0 \\ \sum_{i=0}^{k_0} W[i] & \text{if } k = k_0 \\ 0 & \text{if } k < k_0 \end{cases} \quad (7)$$
$$(k = 1, 2, \ldots, M),$$

where W and k_0 are the inputs of the flooring operation.

4.4 Accelerated Convolution Algorithm in the Case of the Negative Binomial Distribution

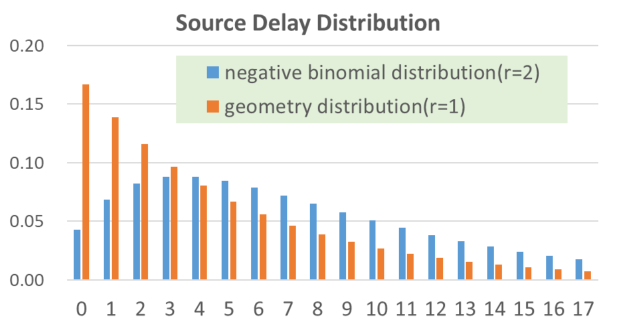

Fig. 2 An example of the geometry distribution ($r = 1$) and the negative binomial distribution ($r = 2$) with these averages at 6.0.

Since we assume the discrete random variables in this paper, the distribution of source delays may also be discrete. As the distribution of source delays $D[k]$, it is desirable to have all the masses on nonnegative domain, most of the masses are on small k domain and having long tails.

The negative binomial distribution, which includes the geometric distribution as a special case, is one of such examples as shown in Figure 2.

The PDF of the negative binomial distribution $D_r[k]$ is as follows:

$$D_r[k] = \binom{k+r-1}{k} p^k (1-p)^r \quad (8)$$
$$(k = 0, 1, 2, \ldots),$$

where p and r are the parameters of the distribution. Note that when $r = 1$, we get the geometric distribution.

Consider the negative binomial distribution as source delays. Then $Z_r[k] = \sum_{i=1}^{k} X[k] \cdot D_r[k-i]$ which is the convolution of $X[k]$ and $D_r[k]$ satisfies the following recurrence:

$$Z_r[k] = \begin{cases} (1-p) \cdot X[1] & (k=1, r=1) \\ (1-p) \cdot Z_{r-1}[1] & (k=1, r=2,3,\ldots) \\ p \cdot Z_1[k-1] + (1-p) \cdot X[k] \\ \quad (k=2,3,\ldots,M, r=1) \\ p \cdot Z_r[k-1] + (1-p) \cdot Z_{r-1}[k] \\ \quad (k=2,3,\ldots,M, r=2,3,\ldots) \end{cases} \quad (9)$$

Utilizing these recurrences, the complexity of $Z_R \leftarrow \mathrm{CONV}(X, D)$ is $O(MR)$, where R is the parameter of the negative binomial distribution. We can also show that the complexity of $O(M)$ is achieved in the case where D is divided into multiple segments each of which corresponds to the part of the negative binomial distribution's PDF. Having a large degree of freedom, this PDF is useful to approximate actual delays.

Note that we must care the truncation error in convolution operations since the output distribution is terminated in finite k. To deal with this, we implement two measures. First, we calculate the $Z_r[k]$ on the domain not only in ranges from $k = 1$ to $k = M$ but in ranges from $k = 1$ to $k = M+b$, since we need to apply the SHIFT(Y, b) operation immediately after the convolution.

4.5 Complexity of the Algorithm

In general case, the complexity of the convolution is $O(M^2)$ and that of other operations in Algorithm 1 is $O(M)$. Thus, the total complexity of the Algorithm 1 is $O(M^2)$, so that the total complexity of the proposed approach is $O(M^2N)$ since we have to apply Algorithm 1 once to each node in the topological order, where N is the number of nodes.

On the other hand, in special cases where source delays are the negative binominal distributions, the complexity can be reduced to $O(M)$ as shown in Section 4.4, so that the total complexity of the algorithm is reduced to $O(MN)$, which means linear (namely very fast) in both of M and N.

5. Computational Experiments

In this section, we evaluate the proposed discrete delay propagation method (**DDP**) from the viewpoint of its accuracy and the calculation time.

5.1 Monte Carlo Simulation Based Approach

To get propagated delay distributions, Monte Carlo simulation (**MC**) can also be used in the following manner. **MC** performs multiple iterations to get histograms of propagated delays in each node. In each iteration, we evaluate nodes in the topological order like **DDP**. However, unlike **DDP**, **MC** regards source delays $d_{i,j}$ as not random variables but deterministic values. As a result, all the propagated delays (X_js) are also deterministic variables. Thus, each node's process with Equation (3) is simpler in **MC** than in **DDP** since **MC** only needs to generate $d_{i,j}$ from $P_{i,j}$ and to perform scalar operations after that.

MC repeats this iteration S times to get histograms of bins $1, 2, \ldots M$ corresponding to the range of discrete vari-

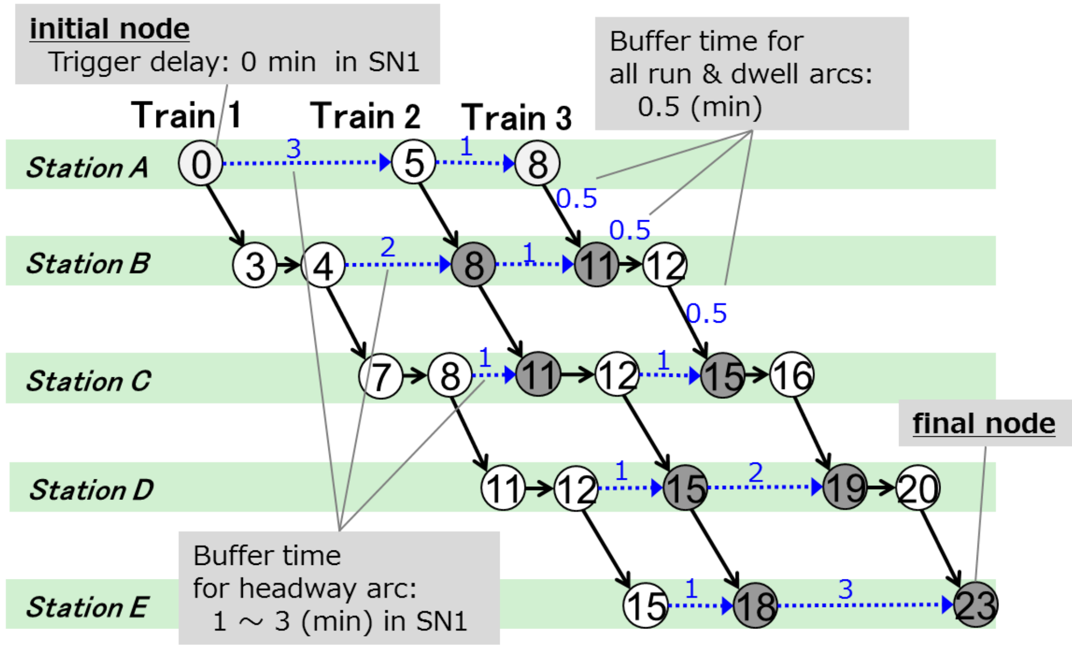

Fig. 3 An example of the network we use in the evaluation. The number in each node shows scheduled time of the node, while the number on each arc shows the buffer time of the arc in SN1.

ables in **DDP**. Then we obtain the propagated delay distribution by dividing frequencies in each bin by S. We set $S = 10,000$ based on preliminary test results.

Note that **MC** does not assume **P_Independence**, so that **MC** can evaluate propagated delays under more natural assumptions.

5.2 Experimental Settings

We used a three-train-five-station network model consisting of 22 nodes as shown in Figure 3. Then we set $M = 720$ ($= 60 \times 60/5$), under assumption that we evaluate delays at most 60 minutes with accuracy of five seconds.

On the network, we evaluate the influence of a trigger delay on the initial node to the final nodes. Then we consider four scenarios. We set trigger delays and buffer times $b_{i,j}$ for each scenario as follows:

- Scenario 1 (SN1): We set no delay on the initial node. Then buffer times for all the headway arcs are from one to three minutes as shown in Figure 3, while those for residual arcs are 0.5 minute.
- Scenario 2 (SN2): We set no delay on the initial node. Then buffer times for headway arcs are values increased toward SN1 by 3 minutes, while those for residual arcs are the same as those for SN1. That is, we assume the situation where we move back the whole sequence of train 2 and 3 by 3 and 6 ($= 3 + 3$) minutes for SN1, respectively.
- Scenario 3 (SN3): We set 10-minute delay on the initial node, and buffer times are same as SN1.
- Scenario 4 (SN4): We set 10-minute delay on the initial node, and buffer times are same as SN2.

We set source delays in the initial node occur at $k = 0$(in SN1 and SN2) or $k = 120$ (in SN3 and SN4) with a probability of 1.0. Then we assume source delays on each arc follow the negative binominal distribution ($r = 2$) with average $\mu_{i,j}$. Here, to make 100 instances for each scenario, we generate $\mu_{i,j}$ from independent discrete uniform distributions in the range from 1 to 24, meaning the range from 5 to 120 seconds.

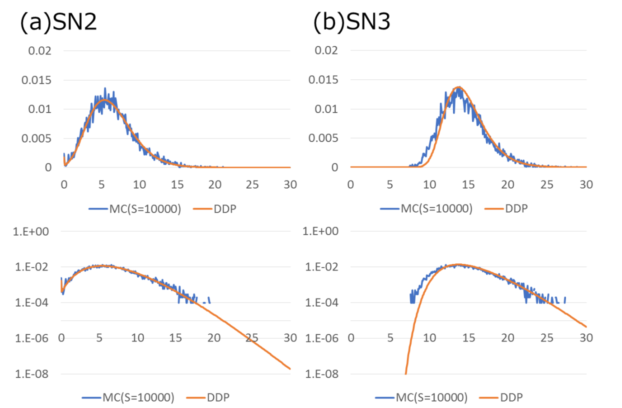

Fig. 4 Examples of final node's propagated delay distributions in **DDP** and **MC** in (a) SN2 and (b) SN3. The horizontal axis in each plot represents the delay time in minutes. The vertical axis is linear in the top figure, while log-scale in the bottom figure for each scenario.

The experiments were all performed on a computer with an Intel Xeon E5-2697v3 2.60 GHz Processor and 264 GB RAM.

5.3 Computational Results

Table 1 shows the compared results of **DDP** and **MC** from SN1 to SN4 by evaluating the propagated delay distribution of the final node.

The third to sixth columns of the table corresponds to the average of mean delay time in minutes (m) and the probability that delays are k minutes or over ($q_k : k = 5, 10, 15$) of the distribution in 100 instances.

In the cases where buffer times between trains are relatively large as in SN2, the result values are almost same in both **DDP** and **MC**. On the other hand, when buffer times are relatively small as in SN1, SN3, or SN4, there are gaps between **DDP** and **MC**. This is because **DDP** assume **P_Independence** while **MC** does not. In case where buffer times are small, **P_Independence** usually does not hold since the correlation in delays from different arcs gets large.

Then Figure 4 shows both linear and log-scale plots of the final nodes' distributions from SN2 and SN3 in **DDP** and **MC**.

As shown in the results in Figure 4, we confirm the curve of **DDP** roughly agrees with that of **MC**, where the probability mass is over 10^{-4}.

However, **MC** cannot calculate the values under 10^{-4} since the number of iterations in **MC** is 10^4, while **DDP** can provide smooth curves when the probability mass is under 10^{-4}. Moreover, **MC** results seem unstable even in the region where the probability mass is slightly larger than 10^{-4}. When we analyze propagated delays, we usually focus on rare events. Therefore, we believe that **DDP** which can evaluate rare events has the advantage over **MC**.

Table 1 Compared results in **DDP** and **MC** from SN1 to SN4. We show mean delays in minutes (m) and the probability that delays are k minutes or over ($q_k : k = 5, 10, 15$). All results are the average of 100 instances.

SN	Method	m	q_5	q_{10}	q_{15}
SN1	**DDP**	6.0	61.0	8.0	0.4
	MC	5.3	49.3	6.2	0.4
SN2	**DDP**	4.1	32.3	3.0	0.1
	MC	4.0	31.3	3.0	0.1
SN3	**DDP**	15.0	100.0	98.7	46.8
	MC	13.9	100.0	93.1	32.9
SN4	**DDP**	9.5	97.9	39.4	3.9
	MC	8.8	93.8	30.7	3.1

Table 2 Comparison of computational time for **DDP** ($T_{\mathbf{DDP}}$) and **MC** ($T_{\mathbf{MC}}$) in seconds when $N = 22, 106, 1002$ and 10004.

N	$T_{\mathbf{DDP}}$ (s)	$T_{\mathbf{MC}}$ (s)	$T_{\mathbf{MC}}/T_{\mathbf{DDP}}$
22	1.11×10^{-4}	1.06×10^{-1}	960.1
106	9.66×10^{-4}	5.29×10^{-1}	548.0
1002	9.71×10^{-3}	5.58	574.5
10004	7.65×10^{-2}	5.64×10^{1}	738.1

5.4 Evaluation of Computation Time

In the experiments above, we found that $N = 22$ is too small to evaluate computation time. Thus, by repeating trains 2 and 3 patterns in the problem shown in Figure 3, we prepared the problem of large N. Table 2 shows calculation time for **MC** and **DDP** with different Ns. The table shows that the calculation time of **DDP** is over 500 times faster than that of **MC** in all Ns.

6. Conclusion

In this paper, we have proposed a novel discrete distribution propagation method (**DDP**) to evaluate delay distributions on railway timetables. We have shown the way of calculating the propagated delay distribution on each event analytically under the assumption of source and propagated delay independence. Then we have shown the complexity of **DDP** is $O(M^2N)$ in the general case and is $O(MN)$ especially in the special cases where source distributions are the negative binominal distributions, where M denotes the number of quantization levels in discretization and N denotes the total number of events. Finally, **DDP** have achieved almost same results as and over 500 times faster than conventional Monte Carlo simulations with 10,000 trials in computation time on test scenarios. In addition to the fast computation time, **DDP** also has the advantage that it can be executed only by easily-accessible operation records such as the average of run and dwell time in each node unlike conventional detailed simulation-based approaches.

On the other hand, **DDP** requires the unnatural assumption of the propagated delay independence. At this moment, we need further analysis regarding the level of error arising from this assumption. Thus, evaluating the effect of the error is and improving **DDP** to cope with the problem need to be established in the future work.

References

[1] Ludolf Meester and Sander Muns: Stochastic Delay Propagation in Railway Networks and Phase-type Distributions, Transportation Research Part B, Vol. 41, pp. 218–230 (2007).

[2] Thorsten Buker and Bernhard Seybold: Stochastic Modelling of Delay Propagation in Large Networks, Journal of Rail Transport Planning and Management, Vol. 2, pp. 34–50 (2012).

[3] Keisuke Abe and Shinji Araya: Train Traffic Simulation Using the Longest Path Method, Transactions of Information Processing Society of Japan, Vol. 27, No. 1, pp. 103–111 (1986).

[4] Yusaku Nagasaki, Motomu Takano and Takafumi Koseki: Train Rescheduling Evaluation and Assistance System with Passengers' Behavior Simulation based on Graph Theory, 2004 Technical Meeting on Systems and Control, IEE Japan, pp. 25–29 (2004).

[5] Norio Tomii, Li Jian Zhou and Naoto Fukumura: An Algorithm for Shunting Scheduling Problems Combining Probabilistic Local Search and PERT, IEEJ Transactions on Electronics, Information and Systems, Vol. 119, No. 3 (1999).

[6] Kohei Ushida, Shingo Makino and Norio Tomii: Increasing Robustness of Dense Timetables by Visualization of Train Traffic Record Data and Monte Carlo Simulation, Proceeding of World Congress on Railway Research (WCRR), Lille, France (2011).

[7] Daisuke Tatsui, Kosuke Nakabasami and Taketoshi Kunimatsu: Predicting Method of Train Delay and Train Congestion Using Neural Network, RTRI report, Vol. 31, No. 10, pp. 29–34 (2017).

[8] Yukihito Nakamura: Simulation Model to Analyze Delay in Commuter Train Schedule, Master Thesis (2004).

[9] Yoko Takeuchi, Norio Tomii and Chikara Hirai: Evaluation Method of Robustness for Train Schedules, Quarterly Report of RTRI, vol. 48, No. 4, pp. 197–201 (2007).

[10] Frank Kirchhoff and Marcin Kolonko: Modelling Delay Propagation in Railway Networks Using Closed Family of Distributions, Available for download at
http://bisec.bjut.edu.cn/en/Events/Colloquium_&_Seminar/201798/15048610866411526_1.html
(last access date: Dec. 27, 2018).

