A Mesoscopic Train Traffic Simulation Algorithm considering Running times of Block Sections

Yutaro WATANABE*, Yasufumi OCHIAI†, Norio TOMII*
* Chiba Institute of Technology
2-17-1 Tsudanuma Narashino, Chiba 275-0015 Japan
E-mail: tomii@cs.it-chiba.ac.jp
† Odakyu Electric Railway Co., Ltd.
1-8-3 Nishi-Shinjuku Shinjuku Tokyo 160-8309 Japan

Abstract
In order to analyze situations of delay propagation in urban railways where trains are running densely, it is required to simulate the detailed movement of trains between stations considering the interaction between trains. At present, two simulation models are known – the macroscopic simulation model and the microscopic simulation model. The macroscopic simulation is not appropriate for this purpose because they only give the arrival and departure times and do not show the movements between stations and the interaction between signaling systems and trains. Thus, the microscopic simulation has to be used. The microscopic simulation, however, has a drawback that the simulation speed is very low. In this paper, we introduce a train traffic simulation algorithm that can simulate not only arrival and departure times of trains but their detailed movement between stations. The algorithm is designed as an extension of the longest path based macroscopic simulation algorithm so that it can deal with running times for each block section. In order to get realistic simulation results, we have introduced an idea to dynamically change the weights of arcs based on an estimation of the signal aspects. Thus, we can simulate trains’ movement between stations reflecting the interaction between consecutive trains.

Key words: simulation, robustness, track circuit, block section, train operation

1. Introduction
In urban areas of Japan, trains are operated very densely. Thus, once a train is delayed due to a small trouble, the delay propagates to many other trains. In particular, in Tokyo area, so called the direct operation, which means trains of suburban areas go directly into subway lines in downtown and vice versa, makes the situation more complicated because a delay tends to propagate to the whole railway network. Thus, railway companies are nowadays very keen to regain punctuality of trains and making every effort to make their timetable more robust (Yamamura 2013, Ochiai 2014, Yamamura 2014). In order to examine effective countermeasures to reduce delays, it is indispensable to evaluate effectiveness of the countermeasures in advance. For this purpose, simulation is usually used. At present, two simulation models are known – the macroscopic simulation model and the microscopic simulation model. Sometimes they are called a discrete model and a continuous model, respectively (Hansen 2014). The difference is the granularity of simulation. In the macroscopic model, only events of trains’ arrival and departure are simulated (Abe 1986, Sato 1980) and we cannot know how trains run between stations. The length of a train, which is a very important factor when we examine the minimum headway of trains is not explicitly considered either. On the other hand, in the microscopic simulation model, trains’ movement is calculated every small time unit based on the Newton’s formula (OpenTrack, Railsys, Janecek 2010). Conditions imposed by signaling systems and length of trains are of course taken into account.
In order to analyze situations of delay propagation, it is required to simulate detailed movement of trains between stations considering the interaction between trains via the signaling systems. This is because trains’ movement is directly influenced by the signaling systems and the interaction between (the location, the speed etc. of) trains and (the aspects of) signals has to be explicitly analyzed. The macroscopic simulation is not appropriate for this purpose and the
microscopic simulation has to be used. The microscopic simulation, however, has a drawback that the simulation speed is very low.

In this paper, we propose a mesoscopic simulation model. This model could be regarded as an extension of the macroscopic model (in this sense, our mesoscopic simulation model is categorized as a kind of the discrete simulation model). A timetable is expressed by an acyclic directed graph. We prepare nodes which correspond to passing of trains at borders of each block section and arcs which express constraints by signals. By calculating the weights of the longest path from the imaginary start node to each node using the longest path algorithm which is used in the longest path based macroscopic simulation, we can calculate the time when a train passes the point (a boundary of block sections) quickly. Thus, the mesoscopic simulation model determines not only the times of arrival and departure of stations but the times when the trains pass the boundary of block sections considering the interaction between the signaling systems and the trains.

The proposed mesoscopic simulation model has a merit that we can easily combine it with the conventional longest path based macroscopic simulation because our mesoscopic simulation model is an extension of the longest path based macroscopic simulation model. This means that we can simulate a part of the timetable (for example, where trains are not running densely and we are not so much interested in the trains’ movement between stations) by the macroscopic simulation and we can simulate the remaining part (for example, where trains are running densely) by the mesoscopic simulation. By this combination, we can realize a simulator which works fast enough and has a required granularity.

Proposed mesoscopic simulation has another merit that it is possible to impose a constraint about the timing of occupation and release of block sections. This means that we can specify the trains’ behavior between stations. In a railway line where trains are running densely, it might be a good idea not to give a complete freedom to the driver about how to drive a train but to impose constraints about the passing times at several points between stations in order to avoid conflicts among trains especially when a train is going to arrive at a station. Our simulator could be used to this purpose.

In the rest of this paper, we first glance the tow simulation models briefly and then we introduce our mesoscopic simulation model. Then we show the results of the numerical experiments using actual train timetable data.

2. Simulation Models – Macroscopic model and Microscopic model

Given the information about the facilities and the timetable and a disturbance, we want to know how disturbance gives an influence to the train traffic. To this purpose, we use the train traffic simulation

Until now, there exist two simulation models. One is a macroscopic simulation model and the other is the microscopic simulation model. The former is sometimes called a discrete model and the latter is called a continuous model (Hansen 2014).

As the macroscopic model, two different models are proposed. One is the event driven model. The event driven model is widely used in industry as a simulator for VLSI design. We choose phenomena whose time of occurrence we would like to know as “events.” Then we put the clock inside the simulator forward with a small time unit and we choose the events which satisfy the conditions. Then we choose one of such events and we “enable” it (namely, the event occurs). When we apply the event driven model to train traffic simulation, we regard trains’ departure and arrivals as events. As for the conditions, physical conditions about necessary times such as minimum running times between stations, minimum dwell times, minimum headways, time for crossover conflicts can be easily expressed (Sato 1980).

The other macroscopic model is the longest path model. In the longest path model, a timetable is expressed by a directed graph, namely a graph which consists of nodes and arcs (Abe 1986). Nodes correspond to events. Arcs express chronological dependency between the two events of the each end of the arc. An arc has a weight which expresses the minimum time necessary for the events of each end of the arc to occur consecutively. By calculating the weights of the longest path from the imaginary start node to each node, we can know the earliest time of those events’ occurrence. When we apply the longest path model to train traffic simulation, we express the events of trains’ arrival and departure as nodes and we set an arc between two nodes if the events corresponding to these nodes have dependency with regards to time to occur. For example, an arc is set between an arrival node of a train and the departure node of the train as a constraint of the minimum dwell time. Another example is an arc which expresses the constraint of the minimum running time. This arc is set from a node of a train’s departure from a station to the node of the train’s arrival at the next station. By setting arcs appropriately, we can express the constrains about capacity such as the number of tracks at
stations, minimum headway time, time for crossover conflict, maximum number of trains allowed to exist between stations (almost the same as the number of block sections between stations) etc. can be realized. The concept of the longest path model is derived from the PERT/CPM (program evaluation and review technique / critical path method) which is a classical method often used in scheduling manufacturing processes for example.

We calculate the weights of the longest path as follows:

1. We topologically order nodes.
2. Following the topological order, we calculate the weight of the longest path.

Here, to ”topologically order” nodes means to give a number $t_i$ (topological order) to a node $i$ so that $t_i < t_k$, if there exists an arc from node $i$ to node $k$ (Sedgewick 2011).

One of the merits of the macroscopic simulation model is that it works very fast compared with the microscopic model. In particular, the longest path model works much faster than the even driven model.

The macroscopic model, however, has a drawback on the other hand. That is, it does not care about the movement of trains between stations. The macroscopic model assumes that trains run ideally between stations. Hence, uniform values are used for (eg.) running times and headway times. For example, it is assumed that trains run with exactly the same time as the technically minimum running time, trains arrive at a station keeping an interval which exactly the same as the technically minimum headway time from the preceding train. But these assumptions do not always hold in the real world. Running times might vary depending on the signal aspects. The headway time might be different depending on the speed of the train. The headway times tend to increase if a train has to stop before it arrives at a station because the track is still occupied by the preceding train and so on. Thus the macroscopic models cannot give a precise simulation results and in order to obtain more precise results, we need to explicitly deal with the signaling systems and the more detailed information such as the aspects of signals, running speed of trains and the length of trains and so on.

The microscopic simulation model calculates the running speed and the position of a train based on the famous Newton’s equation of motion. This model explicitly deals with the detailed conditions which gives influence to trains such as gradients, speed limits imposed by switches, signals, down gradients, curves and so on. Thus, this model offers simulation results in which trains’ detailed movement between stations are considered and nowadays widely used to investigate the robustness of timetables and so on (Čapek 2012, Ercolani 2014, Ochiai 2015). The microscopic model, however, has a couple of deficits such as:

- Simulation speed is much slower than the macroscopic model.
- Detailed data have to be prepared; the data about the trains’ performance such as the acceleration, the deacceleration, the weight, the length), the data about the signaling system such as the positions of the signals, information about the aspects of the signals, the data about the track such as gradient, curves and so on. It is sometimes difficult to prepare these data especially when the (part of the) line is still under construction.
- It is not easy or quite difficult to give a particular indication about trains movement and gets a result in which the indication is realized. In principle, in the microscopic model, trains run as fast as possible. But sometimes we want to give an indication that the train arrives at a specified time. But in the microscopic model, it is not easy to give such an indication.

3. A Mesoscopic Train Traffic Simulation Algorithm considering Running times of Block Section

3.1 Basic ideas

In this paper, we introduce a new simulation model, which we named the mesoscopic simulation model. The mesoscopic simulation model is in a sense an enhancement of the longest path model but in this model not only the arrival and the departure but an occupation and a release of block sections are also considered to be an event. Thus, we can consider the granularity of the model is between the macroscopic model and the microscopic model (this is the reason why we call the model mesoscopic model). Thus, we can know roughly how a train moved between stations. As for the processing time, the mesoscopic model works slower than the original longest path model, but works much faster than the microscopic model.

The mesoscopic model is not a simple enhancement of the longest path model. In the longest path model, weights of the arcs do not change during the process of simulation. On the other hand, in the mesoscopic model, weight of some types of arcs have to be changed dynamically because the running times for a block section vary depending on the signal aspect. In order to realize this, we have introduced a special imaginary arc which expresses a constraint of a
signal aspect. The details of the simulation model is introduced in the next section.

### 3.2 Nodes

There exist five types of nodes as shown in Table 1. The occupation node and the release node are newly introduced in this model, which correspond to an event that the head of a train enters a block section and an event that the tail of a train leaves a block section respectively.

<table>
<thead>
<tr>
<th>Name</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Start</td>
</tr>
<tr>
<td>2</td>
<td>Arrival</td>
</tr>
<tr>
<td>3</td>
<td>Departure</td>
</tr>
<tr>
<td>4</td>
<td>Occupation</td>
</tr>
<tr>
<td>5</td>
<td>Release</td>
</tr>
</tbody>
</table>

#### Table 1: Node

<table>
<thead>
<tr>
<th>Name</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Start node for the calculation of the longest path (imaginary)</td>
</tr>
<tr>
<td>2</td>
<td>Trains’ arrival at a station</td>
</tr>
<tr>
<td>3</td>
<td>Trains’ departure from a station</td>
</tr>
<tr>
<td>4</td>
<td>A train’s head enters a block section</td>
</tr>
<tr>
<td>5</td>
<td>A train’s tail leaves a block section</td>
</tr>
</tbody>
</table>

### 3.3 Arcs

There exist eight types of arcs as shown in Table 2.

<table>
<thead>
<tr>
<th>Name</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Running Minimum running time</td>
</tr>
<tr>
<td>2</td>
<td>Passing Minimum running time</td>
</tr>
<tr>
<td>3</td>
<td>Block section Minimum headway</td>
</tr>
<tr>
<td>4</td>
<td>Signal Minimum dwell time</td>
</tr>
<tr>
<td>5</td>
<td>Arrival Adjustment</td>
</tr>
<tr>
<td>6</td>
<td>Departure Adjustment</td>
</tr>
<tr>
<td>7</td>
<td>Planned Time Planned time</td>
</tr>
</tbody>
</table>

(1) Running arc: We set a running arc from an occupation node of a block section to the occupation node of the block section which is adjacent in the direction of the movement of the train. We give the minimum running time of this train to run this block section as a weight of this arc. The meaning of the running arc is that the adjacent block section is occupied after the time of the weight of the arc. We also set a running arc between two release nodes. The meaning is the same. Figure 1 is an example to show how we set the running arcs.

![Running arcs](image-url)
(2) Passing arc: We set a passing arc from an occupation node of a train to the release node of the block section in which the train existed just before. The passing arc expresses that a train passes the boundary of the two block sections and we set the time needed for the train to run the distance same as the train’s length as its weight. Figure 2 is an example to show how we set the passing arcs.

![Figure 2 Passing arcs.](image)

(3) Block section arc: We set a block section arc from a release node of a block section of a train to the occupation node of the block section of the subsequent train. The weight means the minimum time between the preceding train and the succeeding train. Figure 3 is an example to show how we set the block section arcs.

![Figure 3 Block section arcs.](image)

(4) Signal arc: We set a signal arc from an occupation node of a block section of a train to the release node of the block section of the preceding train. We use the signal arc in order to know the aspect of the signal when a train enters a block section and reflect the speed limits imposed by the signal aspects as shown in Table 2. Under the circumstance of current signaling system, trains have to observe the speed limit until it passes by the block section (more exactly speaking, “until the train receives the information about the aspect of the next signal.” But we have to note that a train can receive the information from the ground beacon when it passes over it).

The aspect of a signal is dependent on the location of the preceding train. So, we first have to detect the location of the preceding train using the signal arcs as clues and from the already calculated times for track occupation and release for the preceding train, we can know the signal aspect for the succeeding train. After we
determine the aspect of the signal, we modify the weight of the running arc so that the decrease of running speed is reflected. Thus, the purpose of a signal arc is two fold: one is a clue to search for the preceding train and the other is a gimmick which enables the occupation and release times of the preceding trains are calculated first. In other words, we respect the signal arcs when we perform the topological ordering (details will be discussed in the next section) and thus the times for the preceding trains are always calculated prior to the succeeding train. Hence, signal arcs do not have a weight. Figure 4 shows an example to set signal arcs.

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Speed limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>G (Green)</td>
<td>(maximum speed in the line)</td>
</tr>
<tr>
<td>YG (Yellow + Green)</td>
<td>75km/h</td>
</tr>
<tr>
<td>Y (Yellow)</td>
<td>45km/h</td>
</tr>
<tr>
<td>YY (Yellow+Yellow)</td>
<td>25km/h</td>
</tr>
<tr>
<td>R (Red)</td>
<td>0km/h</td>
</tr>
</tbody>
</table>

Figure 4  Signal Arcs.

(5) Arrival arc, Departure arc, Dwell arc : An example of a network around a station is depicted in Figure 5. We have to note that at present it is almost impossible to know the exact time when a train arrives at a station and when a train departs from a station in the real world. This is because an arrival time is defined as the time when a train has stopped and a departure time is defined as the time when a train has begun to move and both of them are not recorded at present (technically, it is of course possible if you capture the image of trains by a camera or if you record the motion of a train and so on). Hence, at present, arrival times and departure times are calculated from the occupation and the release time of the block section around the station. In our simulation, we estimate the departure and the arrival times based on the similar idea.

In the network, we prepare an arrival node and a departure node which correspond to a train’s arrival and a departure respectively (if a train passes a station, we prepare a passing node instead of an arrival node and a departure node). We set an arc (an arrival arc) from the release node of the prior block section of this train to the arrival node with a weight which corresponds to the running time until the train stops. Likewise, we set an arc (a departure arc) from the departure node to the occupation node to the adjacent block section of this train with a weight which corresponds to the running time of this train until the adjacent block section is occupied (namely, until the head of the train arrives at the border of the block section). In order to express the constraint of the dwell time, we set a dwell arc from the arrival node to the departure node with a weight which corresponds to the minimum dwell time of the train at this station.
(6) Planned time arc: We set a planned time arc from the start node to each node. We set the planned time as the weight of this arc. For arrival nodes and departure node, the planned times are specified in the timetable whereas for other nodes (occupation nodes and release nodes), planned times are not specified in the timetable. If you do not need to specify the times of the track occupation and release, you do not need to set planned time arcs to these nodes. But by setting arcs to these nodes and set an appropriate value as a weight, you can get a simulation result in which these settings are observed. One way of thinking is that a driver has a complete freedom to drive his/her train between stations and it is all right if he/she just observes the arrival times and the departure times. But another (and probably not yet widely approved) way of thinking is that we impose constraints even for the passing times for some points between stations. We believe in railway lines, where trains are running densely, it is a good idea to impose such constraints to prevent trains from being compelled to stop between stations and being the train delayed.

3.4 Example of a network

We show an example to show how we construct a network for two trains which run consecutively in Figure 6. The dotted lines are signal arcs. Weights of arcs and planned time arcs are not shown to avoid the figure becomes too complicated.
3.5 Simulation algorithm

(1) Input
The inputs to the simulator are as follows:
- Planned timetable
- Planned time for track occupation and release
- Data about the signaling system
- Data about the block sections
- Data about the track layouts
- Weights for arcs
- Initial delay

(2) Output
The outputs of the simulator are as follows:
- Timing of occupation and release for each block section
- Arrival and departure times of trains

(3) Algorithm
The algorithm of the simulation is as follows:
Step 1: Construct a directed acyclic graph (we call it a network) following the procedure we introduced in the previous section.
Step 2: Conduct the topological ordering to the network to give a topological order to each node.
Step 3: Calculate the longest path from the start node to other nodes in the network.

The whole structure of the algorithm is the same as the longest path model but there is one difference. That is in Step 3, we have to dynamically change the weight of arcs considering the aspect of the signal running ahead of a train. We explain the procedure using Figure 7.

Let us assume that we want to know the aspect of the signal when a train (blue train in Figure 7) is going to enter Block section B. We know from the data about the signaling system that the signal is red if the preceding train is in Block section B, yellow if it is in Block section C and green if it is in Block section D. In this example, from the times already calculated (because of the signal arcs, it is guaranteed that the times for preceding trains – in this case the release times of the Block section C of the preceding train), we can know that the preceding train is running in Block section C and we can know that the aspect of the signal must be yellow from the signal arc.

![Figure 7 Identifying signal aspect.](image-url)
4. Numerical experiments

We are conducting numerical experiments using actual data. The target railway line is partly quadruple and partly double and trains are running every couple of minutes during rush hours. We show one of the results of our simulation in Figure 8. In Figure 8, only a part of the results (from 6:30 to 10 o’clock) is shown. As the first stage, we want to confirm that our algorithm works correctly. Then, we have to verify how realistic the results are. In order to confirm the exactness of the simulation, we are comparing the simulation results with actual data of train operation records. We are still on the way to this validation but we prepared the data on this context; that is, we are trying to simulate an actual record of one day in the past (2 July 2013).

Thus, we set the weights of arcs in the following manner:

- Running arc, Passing arc, Block section arc: 95 percentile of the actual data for 25 days in the past. In principle, we should use the smallest value, but in order to avoid extraordinary data, we use 95 percentile.
- Dwell arc: actual value of the dwell times of 2 July 2013.
- Planned time arc: actual value of 14 August 2013 (this is because we still do not have planned times of occupation and release for block sections and on 14 August 2013, there is almost no delay at all. So, we think we may well consider trains should run just like the result of this day).

Initial delays were set only at the station where trains begin their journey in this simulator and the delays are set as the same as those of 2 July 2013.

From the numerical experiments, we have confirmed our algorithm works as we intended. Now, we are still comparing the simulation results with actual train operation records and learned that the results are almost the same as the actual records but there are slight differences especially in the intervals between trains. Hence, the delays tend to decrease in the simulation results whereas this is not the case in reality.

![Figure 8 Result of simulation.](image)

5. Conclusions and future works

We have introduced a mesoscopic train traffic simulation algorithm. The algorithm is a kind of an extension of the longest path based macroscopic simulation but nodes which correspond to track occupation and release and arcs which give constraints about the occurring sequence and the minimum interval necessary between two events are newly added. Thus, using this simulator, we can simulate the behavior of trains between stations, which was not possible by the macroscopic simulation model.

One of the merits of the proposed simulation model is that it works much faster than the microscopic simulation model. Another merit is that it can be easily combined with the longest path based macroscopic mode. Thus, it becomes possible to simulate a part of a railway line in focus using this mesoscopic model and to simulate other parts using macroscopic mode. It is very beneficial to speed up the simulation and to save the time and the labor to prepare the necessary data. One more merit of the proposed simulation model is that it is possible to specify the passing time for several points between stations and obtain a simulation result which observes the specification. At present, it is not
widely approved to specify such times because it might be a big burden for drivers. But we believe in railway lines where trains are running densely, it might be a good idea to specify not only the arrival and the departure times of stations but the passing times for several important points between stations in order to avoid trains are compelled to stop before it arrives at a station.

We have implemented our mesoscopic simulation model and confirmed it works as we intended and we learned that our approach is very promising. We continue our work aiming at getting more precise results of simulation.

Acknowledgement

The third author is partly supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (C) 24510199.

References


OpenTrack - http://www.opentrack.ch

RailSys - http://www.rmcon.de


