Proceedings of the 2017 Winter Simulation Conference
W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page, eds.

A MODULARIZED SIMULATION FOR TRAFFIC NETWORK IN CONTAINER TERMINALS VIA NETWORK OF SERVERS WITH DYNAMIC RATES

Chenhao Zhou
Loo Hay Lee
Ek Peng Chew

Dept. of Industrial Systems Engineering & Management
National University of Singapore
1 Engineering Drive 2
117576 SINGAPORE

Haobin Li

Department of Computing Science
Institute of High Performance Computing
A*STAR Singapore
1 Fusionopolis Way, #16-16 Connexis
138632 SINGAPORE

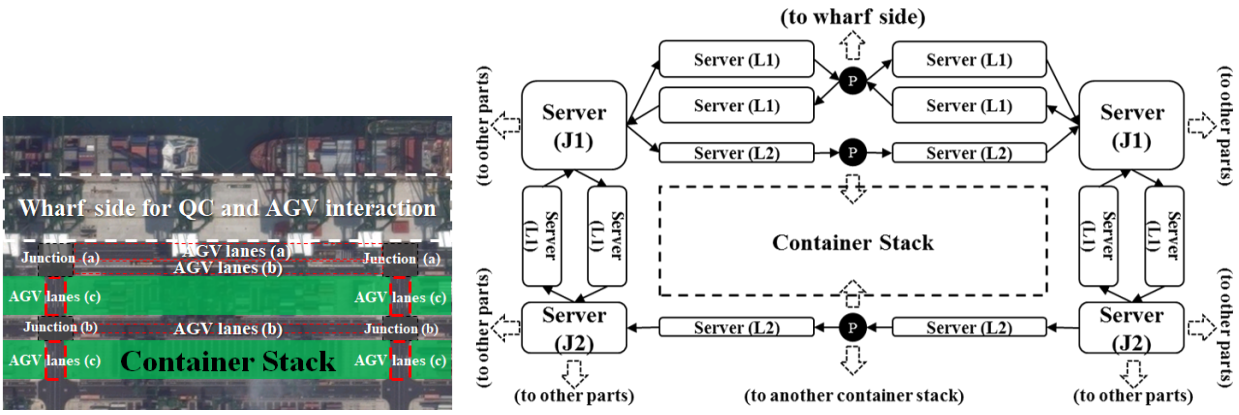
ABSTRACT

There are many design factors affecting the traffic efficiency in an automated container terminal. The traffic efficiency is also dynamically influenced by the nature of the job sequence in the specific container terminal, the number of vehicles deployed, and the respective yard planning strategies. Therefore, it is difficult to analyze such complexity using an analytical queuing model; however, challenges arise in a simulation study such as how to effectively model the impact of the critical design factors as well as the decision rules. In this study, we model the traffic system as a network of servers that represents both paths and junctions, for which the service rates are dynamically adjusted according to the respective states and decision rules. The model is implemented with O^2DES.Net, an open-structured and modularized modeling framework. Numerical experiments illustrate the effectiveness of the developed models, with an application of the AGV network for an automated container terminal.

1 INTRODUCTION

As the container terminal plays a vital role in modern logistics, its productivity and reliability are always the primary concerns of the terminal operators. The traffic network is one of the key components in a terminal system, as it links between quay side and yard side. There are many design factors affecting the traffic efficiency in a terminal, for example, the layout, dimension and connection of the traffic network between junctions and paths, the length and number of lanes in each path, and the number of vehicles (i.e., AGVs) deployed in the network, etc. Dynamically, the traffic efficiency is also influenced by the nature of job sequence in the specific container terminal, the number of vehicles deployed, and respective yard planning strategies. Therefore, it is difficult to analyze such complexity from an analytical queuing model. For example, Roy et al. (2016) proposed an analytical queuing model to estimate the terminal throughput under traffic congestion. Although the congestion on the path was represented by load-dependent server queues, and a non-linear speed-density relationship, the proposed traffic network was simplified in many details, such as the number of lanes on the path, the movement of vehicles and the interaction between vehicles.

Hence, to study the impact of the design factors and the decision rules, simulation models on traffic networks are needed. Discrete event simulation (DES) has been widely adopted in many event-oriented traffic system studies which focus on vehicle travel strategies and rules. For example, Yang et al. (2004) built two DES models to compare AGV and ALV based traffic systems in the same terminal layout. Due to the different behaviors, the events of the two models are also different. Zhou et al. (2016) proposed a



(a) Portion of conatiner terminal traffic system

(b) Conceptualized model with network of servers

Figure 1: Model a traffic system in container terminal as network of servers.

real-time strategy to control the automated vehicle movement in a mesh-like traffic network in a container terminal. The vehicle determined its next waypoint to travel upon reaching its current waypoint, and certain rules were proposed to prevent and resolve conflicts between any two vehicles. Schroër et al. (2014) described a traffic network between multiple terminals. The nodes were used to represent the terminals and intersections, and the arcs were for the roads. The transporters, such as vehicles, could travel through the network, and encounter delays at intersections. In fact, due to the different focuses, some studies used the simplified network to evaluate the proposed strategies in the large layout, while the other models built detailed traffic systems which were only suitable for small layouts.

In recent years, agent-based simulation (ABS) is commonly used to model individual decision-making and behaviors (Bonabeau 2002), which is also useful in the traffic studies that concentrate on the behaviors of drivers and vehicles, such as Hidas (2002). However, since our study focuses on the control strategy from a system perspective for an automation system, DES is more suitable than ABS. In fact, the major difference between the traffic system with man-driving vehicles and an AGV traffic system is how the individual vehicle is controlled. For the system with man-driving vehicles, the control is decentralized, i.e., the drivers will make their own decision when a conflict is met. On the contrary, an AGV system has a centralized control, i.e., the movement of all vehicles is controlled and coordinated by a central system. To guide all vehicles properly, certain strategies and rules have to be well-defined before the system starts operation. For example, if a vehicle is blocked by another one, the system should decide whether to overtake, wait or do nothing.

In this study, we model the traffic system as a network of servers with dynamic service rates. To be specific, the servers are used to represent both paths and junctions, and the service rates are dynamically adjusted according to the respective states of each vehicle, path and junction, and high-level decision rules of the system. The model is implemented with O^2DES.Net, an open-structured and modularized modeling framework. By balancing the detail level of simulation and complexity of the network, the proposed model can be applied in large scale AGV traffic system while testing different event-based control strategies.

2 MODELING METHODOLOGY

As illustrated in Figure 1, we consider the traffic system in a container terminal by using a "path" to represent either a section of multiple lanes towards the same direction or a junction that connects multiple lanes from and to different directions. The basic intuition is that vehicles spend time in traveling through each of them, as a load is processed by a server. However, in a traffic system the traveling time highly

depends on the congestion of the road. Therefore, we consider a dynamic service rate of the "path-server" which depends on the vehicle density on the path.

Li et al. (2015) introduced a simulation modeling framework, which was referred to as O^2DES (object-oriented discrete event simulation). Li et al. (2017) showed the flexibility of the framework with two models at different fidelity levels which depend on how detailed the operation is described. Based on the O^2DES framework, and inspired by the modularization concept introduced in DEVS (Zeigler 1987, Zeigler et al. 2000), the following network model is introduced.

2.1 A Single Path as Server with Dynamic Service Rate

Although they may have distinct dimensions in terms of shape, length, and width, different *path*s still share several common properties.

1. Regarding the interaction between a vehicle and a path, it begins at the time when the vehicle enters the path, and terminates when the vehicle exits from the path.
2. Each path is associated with a finite capacity. A path cannot be entered if the capacity is full, i.e., the number of vehicles reaches the capacity.
3. It is reasonable to assume that the traveling time to complete the path, or the traveling speed, is related to the level of congestion. Hence, the traveling time can be represented by the number of vehicles which are currently traveling on the path.
4. Upon reaching the end of the path, the vehicle will either exit to another path which leads to its destination, or leave the traffic network if the destination is reached, unless the specific path is fully occupied, or the arriving destination is blocked due to external factors, e.g., the yard crane at the container stack is busy and thus the working lane is full.

Let i index of a path, we define the static parameters of a path, i.e., the configuration, as

$$C_i^P = \{l_i, c_i, s_i, o_i, v_i(\rho), h_i\}, \text{ in which} \tag{1}$$

l_i indicates the length of the path i from the entrance to the exit;
c_i indicates the capacity of the path i;
s_i denotes the starting point of the path i, i.e., through which the path i is connected from;
o_i denotes the ending point of the path i the set for all outgoing paths, i.e., through which path i connects to;
$v_i(\rho)$ describes the function to calculate the instant speed of each vehicle traveling on the path i, based on the density of vehicles on it denoted by ρ, in the unit of number of vehicles per unit distance;
h_i is the binary variable indicating if the path i is cross-hatched.

To prevent two cross-hatched paths connected together, it is reasonable to set the constraint that for any path i,

$$\nexists i_1, i_2, \text{ such that } o_{i_1} = s_{i_2} \text{ and } h_{i_1} = h_{i_2} = 1. \tag{2}$$

Let j index of a vehicle, we then define the dynamic properties of any path i, i.e., the state, as

$$S_i^P = \left\{A_i, B_i, m_i, k_i, \left\{d_{i,j}\ \forall j \in A_i\right\}, \left\{t_{i,j}\ \forall j \in A_i\right\}, \left\{x_{i'}\ \forall i' \in \{i\} \cup O_i\right\}, \left\{e_{i'}\ \forall i' \in O_i \cup O_i^2\right\}\right\}, \tag{3}$$

in which O_i is defined as the set of all outgoing paths, i.e., $O_i \equiv \{i' \mid s_{i'} = o_i\}$; and O_i^2 is defined as the set of second consecutive paths, i.e., $\bigcup_{i' \in O_i} O_{i'}$. All state parameters are defined as following:

A_i is the set containing all index of vehicles that is currently traveling on the path;
B_i contains the index of vehicles completed traveling and waiting at the path to exit;
m_i records the last time stamp when the S_i is updated;

$d_{i,j}$ indicates the traveled distance of vehicle j on path i;
k_i records if the vehicles are stuck on the path due to zero vacancy at the consecutive paths;
$t_{i,j}$ denotes the time when the vehicle j is expected to complete its traveling on path i;
$x_{i'}$ is a binary variable to indicate if the end point of path i' is exit-able if it is the destination of the vehicle;
$e_{i'}$ is a binary variable to indicate if the path i' is enter-able, i.e., has vacancy to take any incoming vehicle that continues traveling.

There are four input events that allow external components to trigger state changes of a path, which are defined as following:

- $\alpha_i^{(P1)}(j)$, $\forall j \notin A_i \cup B_i$, is the event that a vehicle j enters the path i;
- $\alpha_i^{(P2)}(i',x')$, $\forall i' \in \{i\} \cup O_i$, is the event to update whether a vehicle is able to exit from the path i' if its target is completed upon reaching the end;
- $\alpha_i^{(P3)}(i',e')$, $\forall i' \in O_i \cup O_i^2$, is the event to update whether a following path i' has vacancy to accommodate any vehicle from path i that continues traveling upon reaching the end, indicated by the binary variable e';
- $\alpha_i^{(P4)}$, is the event to reset the path by removing all vehicles in the path and release any locks incurred by traffic congestion. Note that, this event is for experimental purpose only, as it will never happen in reality.

Meanwhile, three output events are defined as interfaces for the path i to trigger events of external components.

- $\gamma_i^{(P1)}(j)$, is the event on exiting, which is triggered once there is a vehicle j exiting the path i;
- $\gamma_i^{(P2)}(S_i)$, is the event on vacancy change, which is triggered whenever the vacancy of the path i is changed;
- $\gamma_i^{(P3)}(S_i)$, is the event on path locking, which is triggered whenever a vehicle is stopped from exiting the path due to zero vacancy at the consecutive path.

Note that, as output events, although the parameters are specified as the information which the path i provides, the events are not specifically defined as its purpose is to connect to other components (e.g., $\forall i' O_i$) at a higher level modeling.

We now focus on seven events defined to model the activities of vehicles traveling on the path i, in term of internal transition that connects the input and output events. The seven events include four input events, and three internal events triggered by them. The details are described as following, meanwhile, the triggering relationships among the events are illustrated in Figure 2. Note that, in the following description τ is used to indicate the clock time in the simulation.

$\alpha_i^{(P1)}(j)$ It updates the positions of each vehicle as

$$d_{i,j'} \leftarrow d_{i,j'} + v_i\left(\|A_i \cup B_i\| / l_i\right) \cdot (\tau - m_i), \ \forall j' \in A_i, \tag{4}$$

and set $A_i \leftarrow A_i \cup \{j\}$ and $d_{i,j} \leftarrow 0$, followed by an immediate execution of the internal event $\beta_i^{(P1)}$ and the output event $\gamma_i^{(P2)}(S_i)$.

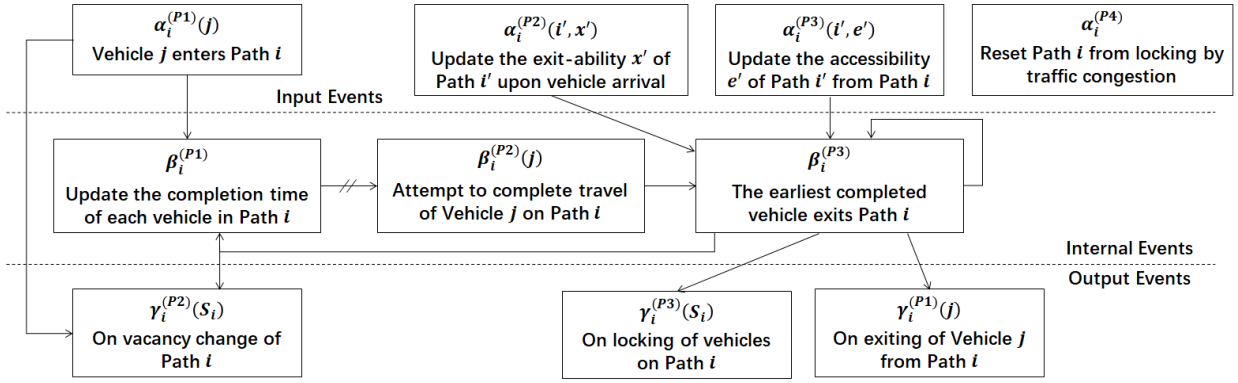


Figure 2: The event-diagram for a modularized Path i .

$\alpha_i^{(P2)}(i',x')$ It updates $x_{i'} \leftarrow x'$, then triggers an execution of $\beta_i^{(P2)}$.

$\alpha_i^{(P3)}(i',e')$ It updates $e_{i'} \leftarrow e'$, then triggers an execution of $\beta_i^{(P2)}$.

$\alpha_i^{(P4)}$ It updates the state as $A_i \leftarrow \emptyset$, $B_i \leftarrow \emptyset$, $m_i \leftarrow \tau$, $e_{i'} \leftarrow 1\ \forall i' \in O_i \cup O_i^2$, and $k_i \leftarrow 0$. Note that $x_{i'}$ remains unchanged as it is not an impact of traffic congestion.

$\beta_i^{(P1)}$ It denotes the internal event that updates the completion time of each vehicle based on the dynamically adjusted speed, due to the state change of path i. Specifically, for each $j \in A_i$, the event updates

$$t_{i,j} \leftarrow \tau + \frac{l_i - s_{i,j}}{v_i\left(\|A_i \cup B_i\| / l_i\right)}, \tag{5}$$

and schedules an event $\beta_i^{(P3)}(j)$ at time $t_{i,j}$.

$\beta_i^{(P2)}(j)$ It denotes the internal event that attempts to complete the traveling of the vehicle j in the path i. The event is aborted if $\tau \neq t_{i,j}$. Otherwise, it updates $A_i \leftarrow A_i \setminus \{j\}$ and $B_i \leftarrow B_i \cup \{j\}$. Then, after updating the positions of each vehicle using (4), it triggers an execution of $\beta_i^{(P3)}$.

$\beta_i^{(P3)}$ It denotes the internal event that the earliest vehicle that completed traveling in the path i exit from it. The event identifies j as the first element in B_i.
We let i' denotes the next path the vehicle j is to enter, and $i' = 0$ if the vehicle reach the destination at the end of the path; and let i'' denotes the second next path the vehicle j is to enter, and $i'' = 0$ if its target is reached at the end of current or next path.
The logic flow as described in Figure 3 is used to decide if the vehicle j is able to exit from the path i. The same flow is to determine if the path i is locked because of congestion, $k_i \leftarrow 1$ if it is "Locked" and 0 otherwise.
Then, in the case of "Exit", $B_i \leftarrow B_i \setminus \{j\}$, the output event $\gamma_i^{(P1)}(j)$ is triggered for execution, and the event itself, i.e., $\beta_i^{(P2)}$, is called for a recursive execution. Subsequently, in the case of "Locked" or $\|A_i \cup B_i\| = 0$, $\gamma_i^{P3}(S_i)$ is triggered.

2.2 The Traffic System as Network of Paths

To configure the static properties of a traffic system, we use Π as the set of index for all paths involved, J as the set of index for all vehicles considered, and we assume the routing rule for each vehicle j is known

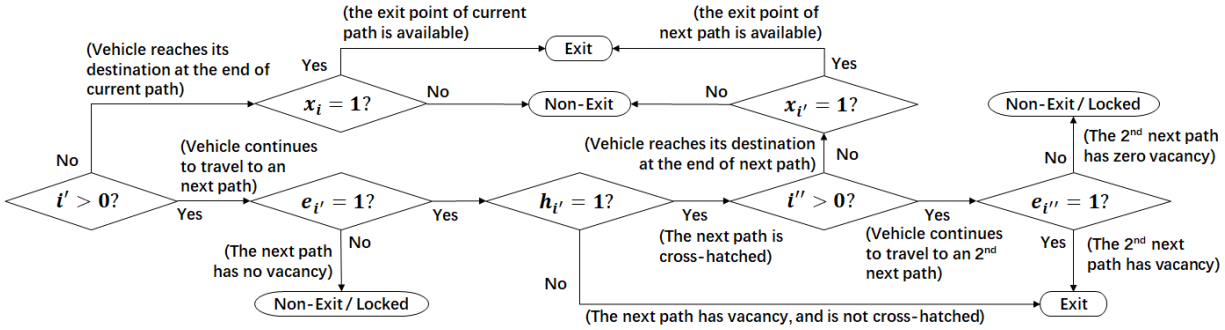


Figure 3: The logic flow to decide if a vehicle j is able to exit from Path i.

as

$$i \leftarrow \delta(j,s), \tag{6}$$

implying that path i is to be taken by vehicle j after it exits from point s. For convention, we let $i = 0$ if the vehicle j has reached its destination at s, and thus exit from the traffic system. In practice, this routing rule can be set based on the Dijkstra's algorithm based on given origin and destination, or by a predetermined routing list for each moving vehicle. Therefore, the static properties of the traffic system can be defined as

$$C^T \equiv \left\{\Pi, \left\{C_i^P \ \forall i \in \Pi\right\}, \delta(j,s)\right\}. \tag{7}$$

Considering the dynamic properties, besides the state of each path i, we also need to consider the vehicles that are departing to travel through the network, but have not entered into any path due to the zero vacancy at the first path on its route. We denote the ordered set of departing vehicles queuing for a path i as Q_i. Then the state of the traffic system is defined as

$$S^T \equiv \left\{\left(S_i^P,\ Q_i\right)\ \forall i \in \Pi\right\}. \tag{8}$$

There are three input events defined at the traffic system level, as following:

- $\alpha^{(T1)}(j,s)$ is the event for a vehicle j to call for departing in the traffic system from point s;
- $\alpha^{(T2)}(s,x)$ is the event to update the state of a point s as x, a binary variable that indicates if it is available for a vehicle to arrive at its destination and exit from the traffic system.
- $\alpha^{(T3)}$ is the event to reset the traffic system from deadlocks.

Correspondingly, three output events are defined as interfaces to trigger external events.

- $\gamma^{(T1)}(j,s)$ is triggered once a vehicle j is departed from point s;
- $\gamma^{(T2)}(j,s)$ is triggered once a vehicle j exits the traffic system from point s;
- $\gamma^{(T3)}\left(S^T\right)$ is triggered once a deadlock is detected.

The internal transitions are defined by the three input events, together with four internal events, and their interaction with the path-level interface events, which are described below. An overview of the triggering relationship is illustrated in Figure 4.

$\alpha^{(T1)}(j,s)$ The event identifies $i \leftarrow \delta(j,s)$, then updates $Q_i \leftarrow Q_i \cup \{j\}$, followed by an immediate execution of $\beta^{(T1)}(i)$.

$\alpha^{(T2)}(s,x)$ For all $i \in \Pi$, execute $\alpha_i^{(P2)}(i,x)$ if $o_i = s$; and in the case of $h_i = 1$, execute $\alpha_{i'}^{(P2)}(i,x)$ for all $i' \in \Pi$ with $o_{i'} = s_i$. This is to update the exit-ability of the point s to all relevant paths in the system.

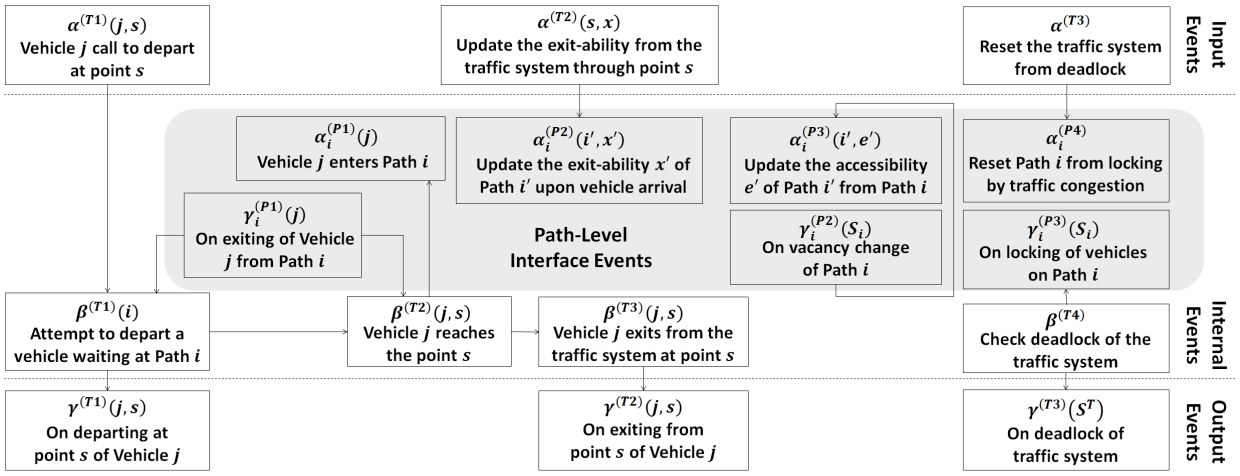


Figure 4: The event-diagram for a traffic network with modularized paths.

$\alpha^{(T3)}$ To reset the traffic system from deadlock, it simply execute the reset event $\alpha^{(P4)}$, and update $Q_i \leftarrow \emptyset$ for all $i \in \Pi$.

$\beta^{(T1)}(i)$ This is the event attempting to depart the vehicle at the start of path i. The event is aborted if $\|Q_i\| = 0$ or $c_i - \|A_i \cup B_i\| = 0$. Otherwise, remove the first element from Q_i and denote it as j_0. Then, execute the internal event $\beta^{(T2)}(j_0, s_i)$, and output event $\gamma^{(T1)}(j_0, s_i)$, followed by the recursive execution of $\beta^{(T1)}(i)$ itself.

$\beta^{(T2)}(j,s)$ This denotes the event when the vehicle j reaches point s. Let $i = \delta(j,s)$ indicates the new path to travel. If $i > 0$, the event $\alpha_i^{(P1)}$ is triggered for execution; otherwise, the internal event $\beta^{(T3)}(j,s)$ is triggered for execution.

$\beta^{(T3)}(j,s)$ Necessary statistics are collected and output event $\gamma^{(T2)}(j,s)$ is triggered for execution.

$\beta^{(T4)}$ This event check if the deadlock exists in the traffic system, which occurs when

$$\sum_{i\in\Pi} \mathbb{1}_{\|A_i \cup B_i\|>0} = \sum_{i\in\Pi} \mathbb{1}_{k_i} > 0. \tag{9}$$

In such a case, the output event $\gamma^{(T3)}(S^T)$ is triggered for execution.

To complete the integrated model, there is a need to bridge the path level output events and the internal events at the system level. The sign of "$\rightarrow$" indicates that the event on the left is triggering the events on the right. Thus, we have

$$\gamma_i^{(P1)}(j) \rightarrow \beta^{(T2)}(j, o_i), \forall i \in \Pi, \text{ and} \tag{10}$$

$$\gamma_i^{(P1)}(j) \rightarrow \beta^{(T1)}(i), \forall i \in \Pi. \tag{11}$$

It implies that once a vehicle exits from a path, we should route it to the next path or arrive at its destination, and attempt to accommodate an incoming vehicle that is waiting at the path. This is to ensure all vehicles move forward to their destinations.

Another connection is to passing the information backward, so as to update the vacancies of paths ahead. Specifically, we have

$$\gamma_i^{(P2)}(S_i) \rightarrow \alpha^{(P2)}\left(i', \mathbb{1}_{\|A_{i'} \cup B_{i'}\|<c_{i'}}\right), \forall i, i' \in \Pi, i \in O_{i'} \cup i \in O_{i'}^2. \tag{12}$$

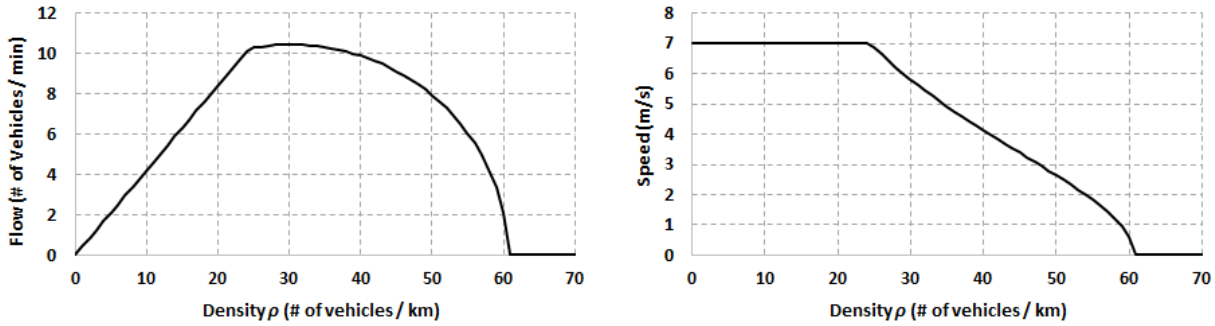


Figure 5: A dynamic speed function for the effect of vehicle density.

Last but not least, whenever there is a lock observed at the path level, the traffic system should check if a deadlock occurs, i.e.,

$$\gamma_i^{(P3)}(S_i) \to \beta^{(T4)}. \tag{13}$$

2.3 Dynamic Speed Function

The critical part of the proposed model is the dynamic speed function of a path i, i.e., $v_i(\rho)$. Although there are several functions discussed in the literature, in this part we attempt to develop a theoretical model with reasonable simplification, so as to test the usability of the model.

We consider the ideal situation in which vehicles can travel at the maximum available speed, and the only constraint is that the average safety distance between two consecutive vehicles should be sufficient for the vehicle to decelerate to fully stop.

Let a be the deceleration of the vehicle, according to the assumption above, the traveling speed v is a function of safety distance d, as

$$v = \sqrt{2ad}. \tag{14}$$

Besides, given the vehicle length L and number of lanes N, the average safety distance can be written as a function of density ρ when it is not negative, i.e.,

$$d = N/\rho - L. \tag{15}$$

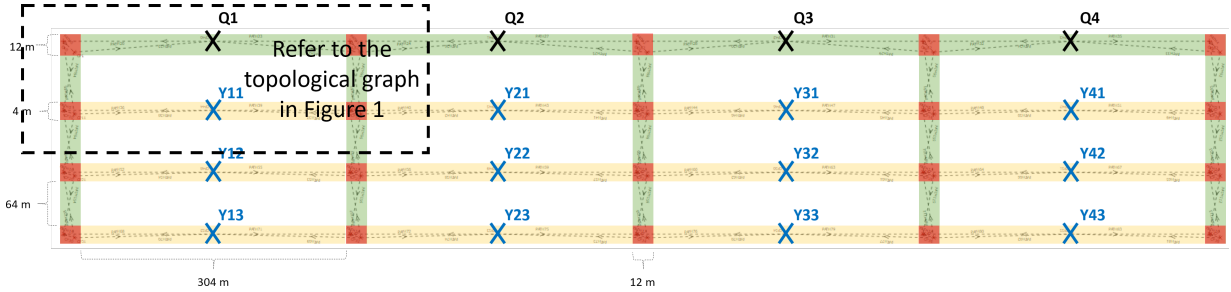
Therefore, combining (14) and (15), and the speed limit $v_{\max}$ the speed function of a path i is represented as following. An example of the dynamic speed function and the resulted vehicle flow curve is illustrated in Figure 5. The function is to be adopted in the numerical experiment in the next section.

$$v_i(\rho) = \min\left(\sqrt{2a\left(\frac{N_i}{\rho} - L\right)^+}, v_{\max}\right). \tag{16}$$

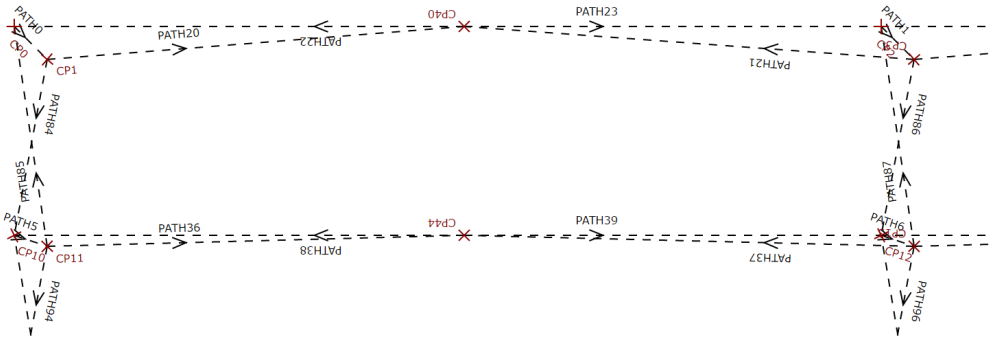
3 NUMERICAL EXPERIMENT

3.1 Experiment Settings

For the numerical experiment, we study the traffic network of a container terminal with 3×4 parallel yard layout, which is usually adopted for a transshipment container port. For transshipment container port, the inner terminal traffic is a critical issue. The dimension of the physical layout is illustrated in Figure 6(a). As shown, the green sections indicate the area of paths with two lanes in each direction (i.e., totally four lanes), and the yellow sections indicate the area of paths with a single lane in each direction. The junctions



(a) Physical layout of the traffic network



(b) A partial illustration of the topological graph

Figure 6: A example traffic network at the container terminal for numerical experiment.

are indicated as red rectangular, and the 4 "work-points" at quay side and 12 points at the yard area are indicated by the black and blue crosses respectively. The "work-point" enables the vehicles to enter into and exit from the traffic system, i.e., move into the "working lanes" which do not obstruct the ongoing traffic.

The topological graph as in Figure 6(b) shows how the physical layout is conceptualized into the entities in the O^2DES simulation model proposed in the last section. Basically, both junctions and multi-lanes paths are modeled as the "path" as in Section 2.1, and are connected via the control points. Note that, the dynamic speed function in (16) is adopted only for multi-lane paths, whereas for the junctions, it is assumed that vehicle travels at half of the maximum speed, throughout the longest Manhattan-distance in the junction (i.e., width + length).

In our experiment, three scenarios as following are to be tested with the shortest path routing rule so as to illustrate the applicability of the proposed model. We implement a travel lock prevention strategy "cross-hatching" which is commonly applied in city traffic. When driving on the road, we can notice some junctions in front of public services are "cross-hatched". It is used to prevent vehicles from entering the path if its exit is blocked so that the path will still be available to accommodate other incoming vehicles from different direction.

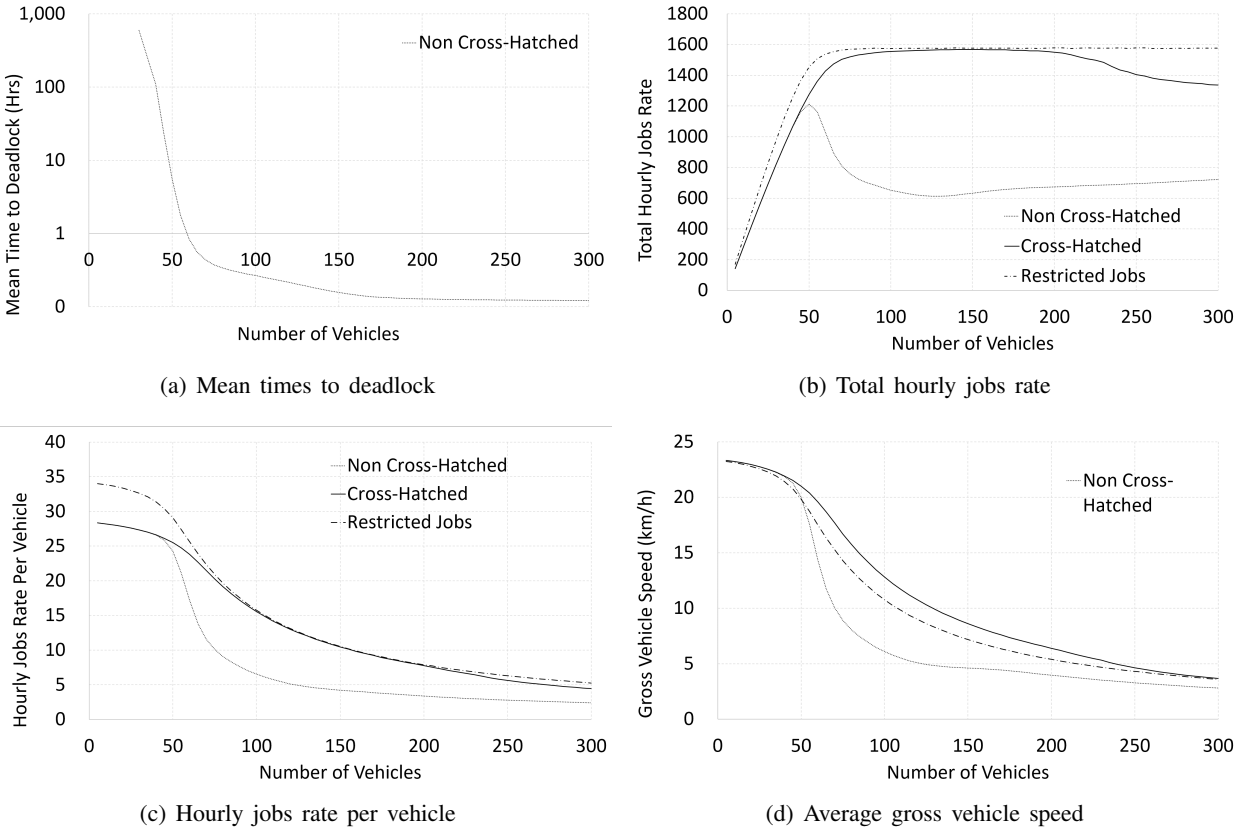
Non-Cross-Hatched In the first scenario, we consider the case no path is cross-hatched, including the junctions. Therefore, it is likely for a vehicle to be stuck at junctions and block the traffic from the other direction. Meanwhile, all targets from either quay-side or yard-area are uniformly sampled from respective groups.

Cross-Hatched The second scenario enforce the cross-hatching at the junctions path, i.e., the vehicle needs to make sure the next path is clear before entering the junction, so as to lower the chance of blocking at the junctions. However, the targets are still uniformly sampled from respective group.

Restricted Jobs (with Cross-Hatching) The last scenario restricts the distances of the traveling jobs, by setting a probability of 95% for sampling a target with at most 1-column difference, whereas with 5% of sampling it other-where. For example, if the vehicle is at "Y22", then with 95% of chance "Q1", "Q2" or "Q3" is sampled; alternatively, if the vehicle is at "Q1", with 95% of chance either of "Y11–Y23" is to be sampled, and the other 5% of chance goes to "Y31–Y43".

In all scenarios, we keep a fleet of vehicles traveling between the quay side and yard side, i.e., after exiting from the target point from one side, the vehicle will enter the traffic network again via the same point and travel to new target point, which is sampled randomly from either side. The fleet size is increased so that we can evaluate the performance of the traffic system at various congestion level. Note that, each scenario is simulated for 30 days and 10 replications, and once the traffic network encounters a deadlock, an immediate resetting, i.e., $\alpha^{(T3)}$ as in Section 2.2, is conducted. The experiments are conducted on a workstation with Intel Xeon E5-2620 CPU and 32GB. For the computation time, each vehicle takes 0.045 seconds for one day simulation in average, which approximately follows a linear trend.

3.2 Result and Discussion



(a) Mean times to deadlock

(b) Total hourly jobs rate

(c) Hourly jobs rate per vehicle

(d) Average gross vehicle speed

Figure 7: Efficiency comparison of traffic network in different scenario.

Firstly, we evaluate the average time for a system to encounter a deadlock, and the results are shown in Figure 7(a) on a logarithmic scale. The result shows that deadlocks start to be observed with 30 vehicles for non-cross-hatched scenario, and no deadlock for the other two scenarios. It provides evidence that the cross-hatching is effective in preventing the deadlocks in the system.

Then, we compare the efficiency of the traffic network in the three scenarios by looking into the multiple indicators, such as total hourly job rate and hourly jobs rate per vehicle. The Figure 7(b) and 7(c) give an overview of the traffic efficiency in terms of jobs completed. Obviously, without cross-hatching at the junctions, the efficiency drops significantly due to the increasing deadlocks and congestions. And when jobs are restricted on its distance, the efficiency is slightly better mainly due to the reduction of time in each journey when the number of vehicles is small. And the results show that the traffic network is saturated when the number of vehicles reaches around 100, and the overall efficiency will drop when more vehicles are added. Therefore, it is economic to keep the number below the saturation. It should be noted that the result of non-cross-hatched scenario after 100 does not have practical meaning since the network will reset when deadlock occurs.

Besides, we can compare the average gross speed, which considers all the traveling time and waiting time due to congestion. It is also an efficiency measure regardless of job distances. The Figure 7(d) provides evidence that restrict job distance does not help in increasing the gross speed with the same number of vehicles.

4 CONCLUSION

In this paper, to balance between the operation details and model complexity, we developed a modularized discrete event simulation model with the network of servers with dynamic service rate, based on the O^2DES.Net framework. In particular, we consider the traffic system in a container terminal by using a "path" to represent either a section of multiple lanes or a junction. Since the traveling time in a traffic system highly depends on the congestion on the road, we consider a dynamic service rate of the "path-server" which depends on the vehicle density on the path. The proposed model can be applied in large scale AGV traffic system and be used to explore different event-based control strategies.

In the numerical experiment, an example of the traffic network is presented, and three scenarios are evaluated with several performance indicators. The experiment demonstrated the applicability of the developed model in answering questions from various perspectives. The model is foreseen to be flexible and extensible to adapt to different speed functions, more sophisticated traffic rules, and vehicle routing principles.

ACKNOWLEDGMENT

This work is supported under the Singapore Maritime Institute research grant: SMI-2015-MA-05.

REFERENCES

Bonabeau, E. 2002. "Agent-based Modeling: Methods and Techniques for Simulating Human Systems". *Proceedings of the National Academy of Sciences* 99 (suppl 3):7280–7287.

Hidas, P. 2002. "Modelling Lane Changing and Merging in Microscopic Traffic Simulation". *Transportation Research Part C: Emerging Technologies* 10 (5):351–371.

Li, H., C. Zhou, B. K. Lee, L. H. Lee, E. P. Chew, and R. S. M. Goh. 2017. "Capacity Planning for Mega Container Terminals with Multi-Objective and Multi-Fidelity Simulation Optimization". *IISE Transactions* (accepted for publication).

Li, H., Y. Zhu, Y. Chen, G. Pedrielli, and N. A. Pujowidianto. 2015. "The Object-oriented Discrete Event Simulation Modeling: A Case Study on Aircraft Spare Part Management". In *Proceedings of the Winter Simulation Conference*, edited by L. Yilmaz, I. Moon, W. K. V. Chan, and T. M. K. Roeder, 3514–3525. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Roy, D., A. Gupta, and R. B. De Koster. 2016. "A Non-linear Traffic Flow-based Queuing Model to Estimate Container Terminal Throughput with AGVs". *International Journal of Production Research* 54 (2):472–493.

Schroër, H. J., F. Corman, M. B. Duinkerken, R. R. Negenborn, and G. Lodewijks. 2014. "Evaluation of Inter Terminal Transport Configurations at Rotterdam Maasvlakte Using Discrete Event Simulation". In *Proceedings of the Winter Simulation Conference*, edited by A. Tolk, L. Yilmaz, S. Y. Diallo, and I. O. Ryzhov, 1771–1782. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Yang, C. H., Y. S. Choi, and T. Y. Ha. 2004. "Simulation-based Performance Evaluation of Transport Vehicles at Automated Container Terminals". *OR Spectrum* 26 (2):149–170.

Zeigler, B. P. 1987. "Hierarchical, Modular Discrete-event Modelling in an Object-oriented Environment". *Simulation* 49 (5):219–230.

Zeigler, B. P., H. Praehofer, and T. G. Kim. 2000. *Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press San Diego, USA.

Zhou, C., E. P. Chew, L. H. Lee, and D. Liu. 2016. "An Introduction and Performance Evaluation of the GRID System for Transshipment Terminals". *Simulation* 92 (3):277–293.

AUTHOR BIOGRAPHIES

CHENHAO ZHOU is Research Fellow for the Department of Industrial Systems Engineering & Management, National University of Singapore. He received his Ph.D. degree from the same department in 2017. His research interest focuses on transportation and logistics in urban city and maritime industry with simulation, optimization to solve complex deterministic and stochastic problems. His email address is zhou_chenhao@u.nus.edu.

HAOBIN LI is Scientist for the Institute of High Performance Computing, A*STAR Singapore. He received his B.Eng. degree (1st Class Honors) in 2009 from the Department of Industrial and Systems Engineering with minor in computer science at National University of Singapore, ; and Ph.D. degree from the same department in 2014. He has research interests in operations research, simulation optimization and designing high performance optimization tools with application on logistics and maritime studies. His email address is lihb@ihpc.a-star.edu.sg.

LOO HAY LEE is Associate Professor for the Department of Industrial Systems Engineering & Management, National University of Singapore. He received Ph. D. degrees from Harvard University. His research interests include production planning and control, logistics and vehicle routing, supply chain modeling, simulation-based optimization, and evolutionary computation. His email address is iseleelh@nus.edu.sg.

EK PENG CHEW is an Associate Professor in the Department of Industrial and Systems Engineering, National University of Singapore. He received his Ph.D. degree from the Georgia Institute of Technology. His research interests include logistics and inventory management, system modeling and simulation, and system optimization. His email address is isecep@nus.edu.sg.