Large scale simulation of a real-time collaborative system for container trucks in the Port of Antwerp

Kenneth Stoop*, Mario Pickavet, Didier Colle and Pieter Audenaert

IDLab,
Ghent University – imec,
Gent, Belgium
Email: kenneth.stoop@ugent.be
Email: mario.pickavet@ugent.be
Email: didier.colle@ugent.be
Email: pieter.audenaert@ugent.be
*Corresponding author

Abstract: In this work, a primary proof of concept of a real-time shared planning system (SPS) for container trucks covering drayage operations in seaports is presented. The aim of such an SPS is to provide a flexible system which allows horizontal collaboration between road carriers with the aim on improving the overall efficiency of the logistic chain. Its impact on the drayage in the port is studied in the context of a large-scale simulation, based on the Port of Antwerp, which models the relevant operations in the logistic chain that handles container transport over roads. In this simulation, interactions on the traffic network will be explicitly simulated by a mesoscopic traffic model. The results show a clear positive impact for carriers joining the collaborative system as compared to the case of individual planning; the time spent in traffic is on average reduced by 13%, meaning less congestion on the road and a potential increase in capacity.

Keywords: container transport; Port of Antwerp; shared planning; horizontal collaboration; truck routing; dynamic scheduling; large scale simulation.

Reference to this paper should be made as follows: Stoop, K., Pickavet, M., Colle, D. and Audenaert, P. (xxxx) 'Large scale simulation of a real-time collaborative system for container trucks in the Port of Antwerp', *Int. J. Shipping and Transport Logistics*, Vol. x, No. x, pp.xxx–xxx.

Biographical notes: Kenneth Stoop received his MSc in Physics and Astronomy from Ghent University in 2019. Presently, he is a PhD student at Ghent University in the Department of Information Technology at the Faculty of Architecture and Engineering. His main research interests are design of algorithms and data structures for complex networking problems.

Mario Pickavet received his MSc and PhD in Electrical Engineering, specialised in telecommunications, from Ghent University in 1996 and 1999, respectively. Since 2000, he is a Professor at Ghent University where he is teaching courses on discrete mathematics and network modelling. He is leading the research cluster on network design, modelling and evaluation, together with Professor Didier Colle. In this context, he is involved in a

large number of European and national research projects, as well as in the Technical Programme Committee of a dozen of international conferences. He has published over 100 international journal articles (*IEEE JSAC*, *IEEE Comm. Mag.*, *Journal of Lightwave Technology*, *Proc. of the IEEE*, *Photonic Network Communication*, ...) and over 300 publications in international conference proceedings. He is a co-author of the book *Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS*.

Didier Colle is a Senior Full Professor at Ghent University since 2022. He was an Associated Professor since 2011 and Full Professor since 2014 at the same university and received his PhD in 2002 and MSc in Electrotechnical Engineering in 1997 from the same university. He is co-responsible for the research cluster on network modelling, design and evaluation (NetMoDeL) inside the IMEC IDlab research group. This research cluster deals with fixed internet architectures and optical networks, green-ICT, design of network algorithms and techno-economic studies. His research is mainly conducted inside international (mainly European), national and bilateral research projects together with the industry. This research has been published in more than 500 international journal and conference articles and has resulted in more than 20 PhDs.

Pieter Audenaert received his MSc in Pure Mathematics (Summa Cum Laude) and PhD with a focus on theoretical aspects of computer science from Ghent University, in 2000 and 2004, respectively. He was a Laureate at the Flemish Mathematics Competition and received a Knuth Reward Check (Stanford University). Currently, he is affiliated with Ghent University/imec and works in the field of networks in its broadest sense: from communication networks and logistic networks, to protein-interactions and social networks. To this aim, he specialises in graphs and algorithms with a focus on applying theoretical results in the field of computer science. This entails, e.g., data-modelling, computational analysis and statistical forecasting but also applications as raytracing and video-analysis or distributed hash tables. He (co-)authored numerous scientific publications, three US patents and one EU patent.

This paper is a revised and expanded version of a paper entitled ‘A real-time collaborative system for container trucks in the port of antwerp: a large scale simulation’ presented at 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), Macau, China, 8–12 October 2022.

1 Introduction

With a worldwide growth of the economy and increase in overseas trade (Sirimanne et al., 2019), ports around the world are continuously facing new challenges in meeting the growing demand in capacity. Roads become more and more saturated, leading to extra costs for transport companies and other stakeholders during their daily operations. Moreover, the inefficiency of how and when trucks are driving from and towards ports causes even more congestion on the roads and terminals, having a huge impact on port operations and the economy in general. There is a need for cross-process communication and collaboration through the logistic chain. Many stakeholders are involved in the

containers transport, but there is little coordination between them (see for example, Carlan et al., 2019).

With the aim on reducing congestion and waiting times and improving their overall efficiency, many terminals have introduced a truck appointment system (TAS), e.g., Huynh et al. (2016). However, within a TAS there is no coordination between road carriers. In our work presented here, the focus lies on the horizontal collaboration between carriers and less so on the vertical collaboration between other stakeholders within the logistic chain, such as forwarders, terminal operators, shippers, etc. We will propose a shared planning system (SPS) for road carriers. The basic idea behind the proposed shared planning system (SPS) is that as opposed to constructing a full planning for the whole day beforehand, the planning will be dynamically constructed on-the-go, enabling great flexibility and allowing to take real-time information into account. The fact that the planning is shared allows for participating trucking companies to exchange orders, providing a bigger pool for the algorithm to pick optimal orders from.

As opposed to many qualitative studies, we will propose an explicit form for a shared planning system for road carriers. Moreover, the proposed framework will be validated in a large-scale simulation. The test-case for the presented SPS and the base for our simulation will be the Port of Antwerp. The Port of Antwerp is an interesting case, where the above mentioned problems of inefficiency and lack of cooperation are difficult to be overlooked (Carlan et al., 2019). Additionally, the Port of Antwerp is an import contributor to the Belgian economy; it is the largest port of Belgium, accounting for 4.1% of Belgian GDP, and a total employment of 141,947 full-time equivalent (Rubbrecht, 2022). It is also Europe's second-largest seaport. Any improvements in operational efficiency will therefore have a significant impact on the Belgian economy.

Our contributions thus consist of two main parts. Firstly an explicit form of a primary proof of concept of an SPS in which road carriers can exchange orders. Secondly the validation of this collaborative system in a detailed large-scale simulation, based on the Port of Antwerp. The traffic of trucks within the road network in and around the port will be explicitly modelled by means of a mesoscopic traffic model. A simulation of the flow of container trucks in and around the port will allow us to study different planning strategies and their impact. In this simulation different aspects of the container flow are modelled, keeping in mind the balance between realism and abstraction/efficiency. It will be demonstrated how the presented collaborative system will improve the average time truckers spend to handle a certain number of orders as opposed to the situation without collaboration where each transport company optimises its own planning. To summarise, our main contributions are:

- a flexible, yet simple shared planning scheme for the horizontal collaboration between road carriers is presented
- the collaborative framework is studied in a large-scale simulation, explicitly taking into account traffic and congestion
- the proposed SPS works in real-time, its on-the-go character allows for great flexibility with regards to carriers or single trucks entering or leaving the collaboration.

This article is organised as follows: the following section will cover the relevant literature: Section 3 will give a description of the traffic model that is used and how routing is handled; Section 4 covers the different functionalities present in the simulation

which are needed to model container transport in the port; in Section 5, the planning strategies which are studied in the simulation framework are described, including the shared planning system; in Section 6, the results are presented; finally, Section 7 contains the final discussion and conclusion from this work.

2 Literature review

2.1 Collaboration

Collaboration within the logistic chain and the setting of a port has been the subject of much research. Often a distinction is made between horizontal collaboration and vertical collaboration (Simatupang and Sridharan, 2005). Horizontal collaboration encompasses all cooperative schemes between two or more independent parties belonging to the same level of the logistic chain, such as joint distribution centres. On the other hand, a form of collaboration that is established among stakeholders acting at different levels of the logistic chain (for instance shippers, carriers, and/or customers) is called a vertical collaboration. Vertical collaboration schemes in hinterland chains of seaports have been studied in, i.a., Notteboom (2009) the impact of horizontal and vertical relations in supply chains on the structure of these chains and on the relationships between seaports and the intermodal hinterland as well as the incentives for market players to vertically or horizontally integrate are studied. Chan and Zhang (2011) proposed new concepts of collaborative transportation management and carriers' flexibility. They used a simulation approach, based on a simple supply chain including one retailer and one carrier, to evaluate and optimise the proposed collaborative management.

In this work the focus lies on the horizontal collaboration between road carriers. This type of collaboration has been studied before as well, for example in Islam and Olsen (2014), the authors provide an extensive qualitative overview of possible obstacles for truck-sharing and successful ways to deal with them, based on a number of semi-structured interviews with road carriers. Carlan et al. (2019) conducted a technical-functional analysis for a 'truck guidance system' in the Port of Antwerp. The approach taken by this report consisted of both desk research and interviews with logistics stakeholders having their activity linked with the Port of Antwerp. This qualitative study gave a global overview of a solution in which digital data is provided to the end-users on a centralised platform. In Caballini et al. (2016), the cooperation between carriers is studied in a quantitative way. An objective function is adopted which considers the total carriers' profit which is maximised by suitably combining the import and export trips shared by the carriers involved in the collaboration. Within their cooperative scheme, a compensation mechanism is designed to take into account the competitive nature of the trucking industry and to encourage carriers to share some of their trips. The problem in their work was formulated as a binary linear program and a few cases were evaluated using real data sets from the Italian port of Genoa. The scale of the case studied was however small; the primary case only contained 30 daily trips in total and only 3 road carriers were considered.

With the aim on reducing congestion and waiting times and improving their overall efficiency, many terminals have introduced a truck appointment system (TAS), e.g., Huynh et al. (2016). In order to reduce truck turnaround times, Shao et al. (2022) proposed an interactive truck appointment system. The system allows drivers/planners to

get an estimation of the turnaround time in function of the preferred arrival time when making an appointment. This framework was evaluated in a discrete event simulation model. However, within a TAS there is not necessarily coordination between carriers. Schulte et al. (2017) present a collaborative framework for trucks to be operated within a TAS, with an emphasis on reducing port-related empty truck emissions. The framework was mathematically described as a mixed linear program with an objective function containing transit costs as well as explicit terms for the emissions, based on the multiple travelling salesman problems with time windows. The scale of the instances tested on was again small; the number of trucks considered range from 4 to 50. Side payments within the coalition of collaborating truckers were mentioned in the problem description, however, no explicit allocation scheme of the profits was given and side payments were not studied in their experiments.

Collaboration in transportation and logistics has also been studied from a game theoretic point of view. For example, Do et al. (2021) have examined competition from a pricing perspective and capacity expansion among alliances between shipping lines, by means of an uncertain competition game model. Göthe-Lundgren et al. (1996) consider the optimal allocation of the cost of an optimal route configuration among the customers in the context of a vehicle routing problem. Regarding a collaboration between road carriers, in order for it to be effective, it is required that the costs and/or profits are divided in a fair way between the participants of the coalition, such that each participant has an incentive to stay in the coalition (Agarwal et al., 2009). In Krajewska et al. (2008), the distribution of costs and saving in a horizontal collaboration between carriers is studied using cooperative game theory; a simple allocation method was used, namely the Shapely value. However, efficient algorithms or heuristics for computation of allocations in large collaborations in logistic planning remain to be investigated.

With regard to the adopted methodology in studying the optimisation of drayage operations through collaboration among carriers, and optimisation in logistics in general, the problem is often formulated as a mixed linear program and optimised as such. This however limits the scale of the problems that can be considered. Another, less common, approach is the utilisation of simulations for measuring and testing proposed schemes, e.g., Gracia et al. (2019), Chan and Zhang (2011) and Dai and Chen (2011). Simulations are in general able to mimic complex emergent behaviour, such as traffic jams, which are difficult to capture in a linear program.

As it already might have become clear, the instances typically considered in literature have a rather small scale. In this work, the problem of horizontal collaboration between road carriers will be studied on a very large scale, i.e., approximately 5,000 trucks and 600 road carriers. To this end a heuristic collaborative planning scheme will be proposed which is flexible enough in order to be able to handle last minute changes and unforeseen events. Since we will be considering collaboration on a very large scale, the simulation approach will be the one taken here.

2.2 Traffic modelling

In the field of research of vehicular traffic dynamics and modelling, different methods of replicating reality exist, each with its advantages and disadvantages. A first class of traffic models are the so-called *microscopic* traffic flow models (Gipps, 1981; Yang and Koutsopoulos, 1996; Brockfeld et al., 2004; Lopez et al., 2018); where each vehicle ν is modelled explicitly and has a well defined position $\vec{x}_\nu$ within the traffic network

as well as a velocity $\vec{v}_\nu$ and acceleration $\vec{a}_\nu$. The dynamics of these types of models is governed by a set of coupled differential equations, namely one equation of motion for each vehicle, dependent upon velocities and accelerations of the other vehicles. Solving this dynamic system becomes quite complex (depending on the assumptions made) and, more importantly, comes with a high computational cost, which is the main disadvantage of this class of traffic models. The advantage of microscopic models on the other hand is that they can be made very realistic and are able to reproduce complex traffic phenomena (Treiber et al., 2000). Another advantage of this type of models is that they allow each vehicle to be tracked individually.

A second main class of traffic models are the *macroscopic* traffic flow models (Papageorgiou, 1998; Di Francesco and Rosini, 2014). These types of frameworks are typically obtained by making the transition from discrete variables to continuous ones. They describe the traffic dynamics in terms of macroscopic quantities defined on each road segment i such as the vehicle density ρ_i, the vehicle flow q_i, etc. A macroscopic description of traffic dynamics bears a strong resemblance with hydrodynamics and gas-kinetic models of physics (Richards, 1956), applying partial differential equations between these different macroscopic quantities based on their fluid dynamical counterparts. The main advantage of these methods is that they allow the study of larger traffic systems as they do not simulate each individual vehicle and are generally less complex than microscopic models. The latter is also a disadvantage as single vehicles can not be tracked with these methods. Another disadvantage is that they are less realistic and detailed, due to the fact that they provide a coarse-grained view of reality on a scale that is not completely suited for this approach.

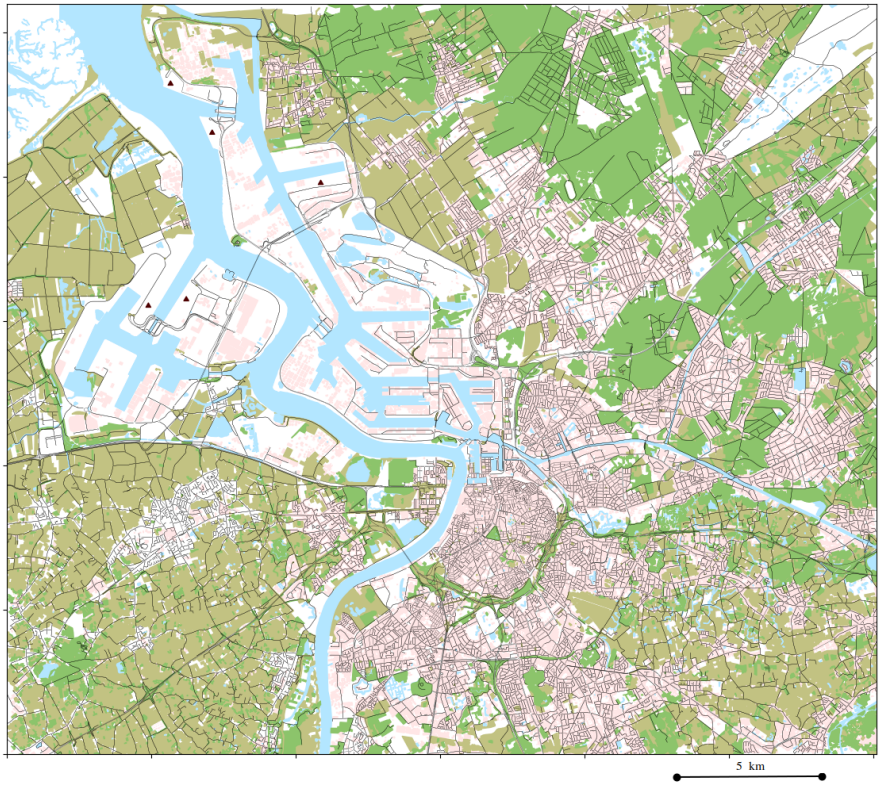
The third class of simulation techniques falls in between the previous two, these are the *mesoscopic* traffic flow models (Ferrara et al., 2018; Eissfeldt, 2004). In general, they do not track vehicles' exact positions, vehicles are present on a certain road segment, but their position within this segment is unspecified. Individual vehicles are moved either according to mean densities on street segments or queueing models (Akamatsu, 2001), where vehicles or traffic volumes are moved without modelling the dynamics inside the segments. In such a queueing model the overall dynamics are governed by the rules of transitioning from one segment to the next. Mesoscopic models allow for a balance between computational efficiency as well as a high degree of realism. Moreover, they work on the scale of individual vehicles, which makes them very well suited for our application. This is why a mesoscopic model will be used in this study, more specifically a queueing model based on the μ-queue model from Eissfeldt (2004). The reason this model is chosen is that it is relatively simple, yet is able to properly reproduce the backward propagation of jam waves, as opposed to other queueing models (Gawron, 1998a, 1998b), where traffic jams remain stationary. The next section will give a detailed description of the model that is used to test the proposed shared planning system.

3 Traffic simulation

At the base of every traffic model lies a network structure that represents the traffic network. The network considered in this simulation is the complete road network in a rectangle of about 30 km $\times$ 25 km around Antwerp and its port, see Figure 1. The

networks consists of $|V| = 20{,}489$ nodes or intersections and $|A| = 46{,}685$ arcs or roads connecting them (only roads where trucks are allowed are included). The traffic network data was obtained from OpenStreetMap Contributors (2020).

Figure 1 The road network (for trucks) around Antwerp and its port (see online version for colours)



The mesoscopic traffic model that is used in this study is based on (state-dependent) queueing theory. In terms of queueing theory, each link of a street network is regarded as a queue, obeying the first-in, first-out (FIFO) principle, i.e., a service device operating at a certain service rate which corresponds to the flow capacity of the link, being the maximum throughput in [vehicles/h] which can be maintained. Queues of vehicles (congestion) occur in the system, whenever the current demand exceeds the flow capacity of a service. As a consequence, vehicles queue up in front of the service device, and experience additional waiting times before being served. The total time a vehicle spends on a link, therefore, equals the sum of the waiting time due to congestion and the service time. Moreover, the service times will depend on the state (i.e., the density) of the considered link. This allows to replicate the phase transition that occurs in real vehicular traffic systems, namely from the free flow phase to the jamming phase, where a jam or *shock wave* propagates backwards through the system.

The model presented here is based on the work in Eissfeldt (2004), the *μ-queue* model, although some additions and adaptations are made. For a detailed description

of this model, we refer to Appendix. The efficiency of the mesoscopic model allows to simulate ten thousands of vehicles in a large network with hundreds of thousands of arcs/roads, while being able to reproduce traffic jams and track individual vehicles. To take into account the current load on the traffic network when calculating optimal routes and avoid traffic jams, a delay metric or an additive resistance is added. In the simulation, this is done in the form of modifying the weights w_i of the arcs from simply the time it would take to traverse it, $\frac{L_i}{v_i}$, to a more general form:

$$w_i = \begin{cases} \frac{L_i}{v_i} & \text{if } n_i < n_i^{jam} \\ \frac{L_i}{v_i} \frac{n_i(c_i - n_i^{jam})}{n_i^{jam}(c_i - n_i)} & \text{if } n_i \geq n_i^{jam} \end{cases} \tag{1}$$

where n_i is the number of vehicles on road i and is taken to be $\min(n_i, c_i - 1)$ to avoid infinite numbers. The first case corresponds to the situation of free flow, whereas the second case is the travel time in the jammed phase where the weight is derived from the average speed as given by the fundamental diagram. In the simulation, the exponential moving average (EMA) $\widetilde{w}_i(t)$ is used for the weights, with a scale parameter $\tau = 10$ min.

For more details on the traffic simulator that was built, we refer to Appendix.

4 Simulation of container transportation in the port

To simulate the dynamics and evolution of the container transport by trucks in the port, some extra functionalities have to be modelled and implemented. Below we will give a description of the different elements and aspects that are of importance in studying the flow of container transport (by trucks) in the port and how they are represented.

4.1 Terminals

The Port of Antwerp has five main container terminals (see Figure 2). In Table 1 their annual capacity is given in TEU (twenty-foot equivalent unit). The containers that are typically transported by a semi-trailer truck are double in length (FEU, 40-foot equivalent unit), which is a standard size, meaning that these numbers represent double the amount of these 40-foot containers. The total amount of containers handled in the Port of Antwerp in 2019 is 11,860,204 TEU, 58% of which is handled by trucks, 34% by barge and 8% by rail (Port of Antwerp, 2020). This means that 3,439,459 40-foot containers were handled by trucks in 2019. We thus assume that all orders consist of a 40-foot container, which is by far the most common type. If there are any 20-foot container orders, it is assumed that they are combined on one trailer. All of the terminals are opened (landside) 24/5, except for Antwerp Container Terminal which opened 5 days from 6:00 to 21:15. So there is a total of about 250 operating days in a year, meaning that on average $N_o = 13{,}758$ (40-foot) containers are handled each day. Assuming that the number of containers processed in each terminal is proportional to the respective capacity, this can be converted to the average daily processed number of containers by trucks for each terminal, see the last column in Table 1.

Figure 2 The locations of the five container terminals in the port (see online version for colours)

Source: Port of Antwerp (2020)

Table 1 The five container terminals in the Port of Antwerp, their capacities and the estimated average daily throughput

Terminal	*Capacity (TEU)*	*Daily throughput (FEU)*
MSC PSA European Terminal (MPET)	9,000,000	7,327 (53%)
DP World Antwerp Gateway Terminal	2,500,000	2,035 (15%)
PSA Antwerp Europa Terminal	1,800,000	1,465 (11%)
PSA Antwerp Noordzee Terminal	2,600,000	2,117 (15%)
Antwerp Container Terminal	1,000,000	814 (6%)

The terminals operate in similar ways although there is a difference in how much of the truckflow inside each terminal is automated. Some terminals have an online time slot booking system (or TAS), however, for the case of Antwerp, these time slots are not binding and trucks can arrive at any time in the day. On a high level, terminals can be seen as a system which processes trucks. The time a truck spends in the terminal depends on many things, the most important factor being the current load on the terminal, i.e., how many trucks it has to process at a given time. To model this behaviour, terminals will be represented by a G/M/1 queueing model, in which elements are processed with times that follow an exponential distribution, with an average service rate μ. The service rate is a measure of how many trucks are processed per unit of time and thus of the capacity of that terminal; it is assumed that $\mu \propto$ capacity (see Table 1). The time at which a truck ν exits the terminal t^{ν}_{exit} is given by

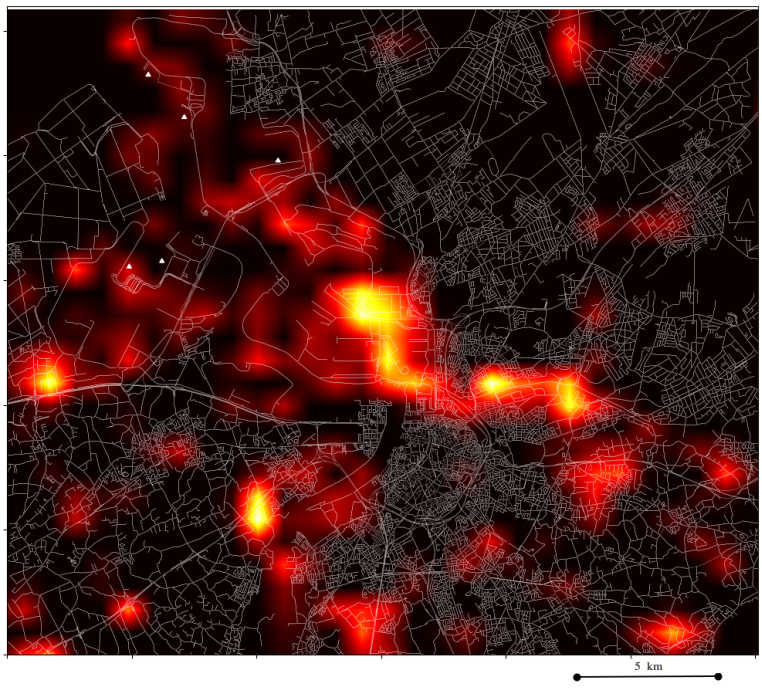
$$t^{\nu}_{\text{exit}} = \max(t, t^{\nu-1}_{\text{serv}}) + T^{\nu}_{s} + T_h \tag{2}$$

where $t_{\text{serv}}^{\nu-1} = \max(t, t_{\text{serv}}^{\nu-2}) + T_s^{\nu-1}$ is the service time of the previous truck that entered the terminal and $T_s^{\nu} \sim \text{Exp}(\mu)$ is the service time (time between services). This first part of equation (2) is a direct consequence of the G/M/1 queue. The last term $T_h \sim \mathcal{N}(h, \sigma)$ represents the extra handling time due to different kinds of operations inside the terminal (multiple checks, waiting in the parking area for the container to be loaded etc), which we assumed to follow a normal distribution. Note that due to this last term, trucks will not necessarily exit the terminal in the same order in which they arrived. The parameters are set to $1/\mu = 2.0 \cdot (9{,}000{,}000/C)$ s, with C the annual capacity of said terminal, $h = 1{,}800$ s and $\sigma = 200$ s.

4.2 Orders

In the context of this study, an order is a 40-foot container which has to be picked up somewhere and has to be dropped off in another location. One of these locations, either the pick-up or drop-off, will be a terminal, the other locations will be somewhere in an industrial area in the hinterland. There are thus two types of orders, *drop-off*, meaning a container is picked up somewhere and dropped off at a terminal, and *pick-up*, the reverse. It is assumed that roughly equal amounts of the orders are pick-up or drop-off (Port of Antwerp, 2020) and that the handling times in the terminals are similar for both.

Figure 3 A heatmap showing the resulting distribution of drop-off or pick-up locations that are not located at one of the five terminals (see online version for colours)



Detailed data on the origin-destination pairs of containers in the Port of Antwerp is not available. In order to roughly approximate potential drop-off or pick-up locations, map data from OpenStreetMap Contributors (2020) was used. On these maps, different areas are classified according to the main activities or characteristics of these areas (building, forest, waterway, etc.). All patches that are classified as 'industrial' are filtered out and all road segments that fall inside one of those industrial patches are used as potential locations for container pick-ups or drop-offs; see Figure 3 for the resulting distribution. Finally, orders are generated as follows: a random terminal is chosen (weighted by the capacities), a random industrial road segment is chosen (with a probability proportional to its length), and finally with 50% probability, the order is set to either drop-off or pick-up. During the simulation, when an order is loaded or unloaded on a non-terminal location, a delay is added to simulate the time needed to carry out this operation, drawn from a normal distribution $\mathcal{N}(\mu_l = 1{,}800 \text{ s}, \sigma_l = 200 \text{ s})$.

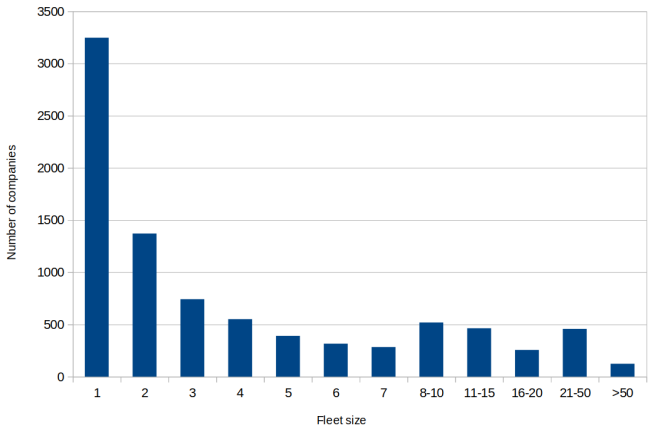
4.3 Trucking companies

The players that handle the orders and are responsible for the majority of the container transport over land are the trucking companies, each having their own fleet of trucks. Figure 4 presents a snapshot of the road transport market in Belgium. Note that of a total of approximately 8,700 road transport companies, around 3,200 (37%) are companies with one vehicle. A similar distribution of trucking companies and fleet sizes that operate in and around the Port of Antwerp will be assumed. The total number of trucks operating in the port will be set to $1/3$ of the total number of orders handled daily (such that every truck handles 3 orders each day, on average), i.e., $N_t = 4{,}586$. The resulting distribution of the number of companies N_i with fleet size i can be modelled by a power law:

$$N_i = N_1 i^{-1.3} \tag{3}$$

The constant N_1 is determined by the condition that the total number of trucks should be equal to N_t, so $N_t = \sum_i^k iN_i = N_1 \sum_i^k i^{-0.3}$ with $k = 50$ a cut-off.

Figure 4 Overview of Belgium trucking companies' fleet size (see online version for colours)



Source: FOD Mobiliteit (2017)

5 Planning strategies

With the framework described in the previous section, a simulation can be set up: orders are allocated to trucks by a certain planning strategy and trucks carry out the orders by visiting the necessary location and terminal. The simulation ends once all orders are processed.

By planning strategy of a trucking company, we mean the way in which pending orders are allocated to trucks which will handle them, for which a multitude of possibilities exist. It is possible to make a complete planning before the start of the day, if all orders are known. A main problem with this is that it is very difficult to make accurate predictions of traffic situations and situations at the terminals and to take them into account in optimising the planning. Predictions being off by only a little bit can have a great impact on the precomputed planning and result in it not being optimal anymore. Another main problem in the same line is that such a method is not very flexible, which is a necessity when serving this many independent road carriers. For example, new urgent orders being placed during the day or trucks/drivers that cannot drive that day for unforeseen reasons are difficult to take into account. Which are things that are bound to happen when such a large amount of trucks and orders are involved. These are the main reasons why on-the-go real-time planning strategies will be considered here. They are very flexible and allow one to anticipate on real-time information on traffic situations, terminal waiting times, etc. This flexibility will be especially important when considering global optimisation in the proposed SPS, see Section 5.2 below.

With 'real-time planning', we mean that orders are assigned to trucks in real-time. Trucks that are inactive (i.e., have no order assigned to them) request an order and receive one. Once they finished this order, they can again request an order.

5.1 Local optimisation: individual planning

The first strategy that will be discussed is *local optimisation*, denoting planning strategies where each competitive trucking company plans for its own orders and trucks, without any collaboration.

A first method of 'planning', to which others can be compared, is *random planning*, where trucks are assigned random orders. This represents the case where trucking companies do not really take any objectives or information into account and just carry out orders on the go. An important factor that can be taken into account in assigning orders to trucks is the expected travel times from the current location of the trucks to the pick-up location of the order. Given a set of pending orders and pending trucks, one can assign (timewise) shortest order-truck pairs to one another. One can see this as $|O_p|$ orders and $|T_p|$ trucks with $|O_p||T_p|$ links between them with a weight representing the travel time between the location of the truck and the pick-up location of the order. Trucks have to be assigned to orders such that the sum of the expected travelling times is minimised. This is a well known combinatorial optimisation problem for which good algorithms exist (Fredman and Tarjan, 1987; Ramshaw and Tarjan, 2012). The complexity of the *Hungarian algorithm* which solves this problem exactly, amounts to $O(|O_p||T_p|\min(|O_p|,|T_p|))$. This is, however, too slow for our application and here an approximate technique will be used.

Assignments with the minimal travel times are picked in a greedy fashion, the pseudocode is given in Algorithm 1. The time-complexity of this approximate algorithm is $O(|O_p| + |T_p|)$. It works by taking the set, O_p or T_p, with the smallest number of elements in it and going over them one by one, and assigning the closest truck/order to it. This is done by doing a Dijkstra search from one location to a set of possible locations (ClosestLeaf and ClosestRoot in the pseudocode), which can be done efficiently, resulting in min$(|O_p|, |T_p|)$ calls to the Dijkstra algorithm.

Note that a lot of orders/trucks can have the same location, namely one of the five container terminals. That is why the maps L_o and L_t are used, for each location they contain a linked list (LL) with the orders/trucks with the same locations. These LL are randomised for reasons of fairness. Equally important: the order in which the trucks and orders are iterated through is randomised. This will ensure no truck will be favoured over another and more importantly, no trucking company gets an advantage over another in the SPS (see below).

Algorithm 1 Assign-Orders-Local()

$O_p = \{...\}$ ▷ Set of pending orders
$T_p = \{...\}$ ▷ Set of pending (inactive) trucks
$A = \{(.,.),...\}$ ▷ Empty map of (order, truck) pairs that will be assigned
if O_p **not** empty **and** T_p **not** empty **then**
  if $|O_p| \geq |T_p|$ **then**
    $L_o = \{(., LL[\,]),...\}$ ▷ Map from pick-up locations to LL of orders
    for each truck t in T_p **do**
      $a \leftarrow$ ClosestLeaf(t.location(), L_o.keys()) ▷ Arc from set with shortest path from location to it
      $o \leftarrow L_o$.get(a).pop() ▷ Closest order to truck t
      A.put(o, t)
      O_p.remove(o)
    T_p.clear()
  else
    $L_t = \{(., LL[\,]),...\}$ ▷ Map from truck locations to LL of trucks
    for each order o in O_p **do**
      $a \leftarrow$ ClosestRoot(o.pickuploc(), L_t.keys()) ▷ Arc from set with shortest path from it to location
      $t \leftarrow L_t$.get(a).pop() ▷ Closest truck to order o
      A.put(o, t)
      T_p.remove(t)
    O_p.clear()
return A

Algorithm 2 Initialise-Master-Set()

$MO_p = \{...\}$ ▷ Master set of pending orders
$Cr = \{(.,.),...\}$ ▷ Empty map of (trucking company, credit)
for each trucking company tc **do**
  $O_p \leftarrow tc$.pendingOrders() ▷ Set of pending orders of tc
  MO_p.addAll(O_p)
  Cr.put($tc, |O_p|$)

Algorithm 3 Assign-Orders-Global()

```
1: MO_p = {...}                                          ▷ Master set of pending orders
2: Cr = {(.,.),...}                                      ▷ Map containing credit for each trucking company
3: A = {(.,.),...}                                       ▷ Empty map of (order, truck) pairs that will be assigned
4: MT_p = {...}                                          ▷ Set of pending (inactive) trucks
5: for each trucking company tc do
6:     if Cr.get(tc) > 0 then
7:         MT_p.addAll(tc.pendingTrucks())
8: if MO_p not empty and MT_p not empty then
9:     if |MO_p| ≥ |MT_p| then
10:        L_o = {(., LL[ ]),...}                        ▷ Map from pick-up locations to LL of orders
11:        for each truck t in MT_p do
12:            tc ← t.truckingCompany()
13:            C ← Cr.get(tc)
14:            if C = 0 then
15:                continue
16:            a ← ClosestLeaf(t.location(), L_o.keys())   ▷ Arc from set with shortest path from location to it
17:            o ← L_o.get(a).pop()                      ▷ Closest order to truck t
18:            A.put(o, t)
19:            MO_p.remove(o)
20:            Cr.put(tc, C − 1)
21:    else
22:        L_t = {(., LL[ ]),...}                        ▷ Map from truck locations to LL of trucks
23:        for each order o in MO_p do
24:            C ← 0
25:            while C = 0 do
26:                a ← ClosestRoot(o.pickuploc(), L_t.keys())   ▷ Arc from set with shortest path from it to location
27:                t ← L_t.get(a).pop()                  ▷ Closest truck to order o
28:                tc ← t.truckingCompany()
29:                C ← Cr.get(tc)
30:                if C > 0 then
31:                    A.put(o, t)
32:                    Cr.put(tc, C − 1)
33:        MO_p.clear()
34: return A
```

5.2 Global optimisation: a shared planning system

Now the matter of *global optimisation* will be discussed, by which we mean a planning strategy that aims at optimising the container transport for trucking companies by collaborating and sharing orders. An important property/constraint to keep in mind is that the strategy should be beneficial for all participating parties in order for it to be successful, i.e., it should be individually rational in a game theoretic sense. Keeping this in mind, the following scheme is proposed: create a *master set* of pending orders in which all pending orders of participating trucking companies are put, all trucks of the participating companies are treated equally. Upon requesting an order, trucks are assigned an order in a similar fashion as in the local Algorithm 1 given in the previous section. The basic idea behind this is that the system now has a bigger pool of orders

from which optimal ones are picked and allocated to trucks, compared to individual trucking companies. This results in companies sharing orders while being profitable for each company using this joint sharing system. An important constraint that is introduced in this strategy is that each company can only get as many orders from the master set as it has put in at the start of the day/simulation; assuming all orders are of equal value; if they are not, it is straightforward to generalise. This will ensure no trucking company can obtain more orders than they initially had. The pseudocode is given in Algorithms 2 (initialisation) and 3 (actual planning).

We thus propose a real-time planning system that is used by multiple different road carriers. In this dynamic on-the-go planning, pending orders are shared between transport companies as a common good which allows for a more profitable planning and allocation of orders to trucks compared to individual planning. Every time a truck is available, it can request a new order from the system, which will return an optimal one taking into account current traffic situations and the current position of the truck. The real-time nature of this planning system inherently allows the use of real-time information, such as current traffic situations, which is done through the shortest-path allocation of orders to trucks which uses the current load on the road network. Moreover, this real-time booking allows for great flexibility, orders, trucks and even trucking companies can join and leave the system without the need of redoing the planning.

6 Results

In this section the results of the simulation and the different planning strategies will be discussed. The simulation is implemented in Java 11.0.7 and the experiments were performed on a computer with an Intel Core i7-8650U CPU @ 1.90 GHz × 8 processor and 16 GB of RAM, under Ubuntu 18.04 x64. The time needed to complete a full simulation depends on the parameters used, for the parameters mentioned in the previous sections this amounts to approximately 220 s. An overview of the parameters used in the simulation can be found in Table 2.

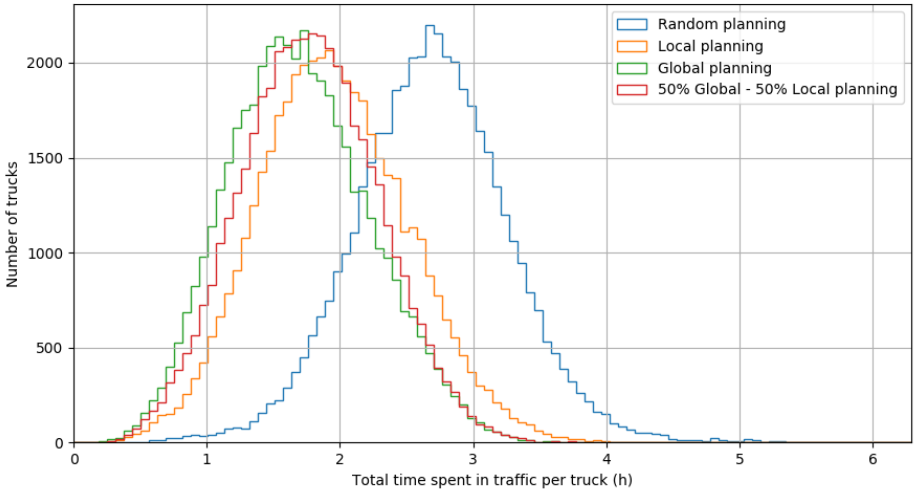
Table 2 The simulation parameters used in the experiments

Traffic model	$v_{\text{max}} = 90$ km/h
	$L_{\text{vehicle}} = 18$ m
	$\tau_{\text{ff}} = 2.5$ s
	$\tau_{\text{jf}} = 3.5$ s
	$\rho_{\text{jam}} = 1/42$ m^{-1} at $v = 50$ km/h
Routing	$\tau = 10$ min
	$\Delta t_{\text{update}} = 5$ min
Terminals	$T_h \sim \mathcal{N}(h = 1{,}800 \text{ s}, \sigma = 200 \text{ s})$
	$1/\mu = 2.0 \cdot (9{,}000{,}000/C)$ s
Container transport	$N_t = 4{,}586$
	$N_o = 13{,}758$

6.1 Comparing planning strategies

To compare the different planning strategies described in the previous section, ten simulations were done for each strategy, with otherwise the same parameters and initial conditions. In Figure 5, the distribution of the times spent in traffic by each truck is shown for four different cases: random planning, local planning, global planning, and a transitional case with 50% of the trucks in the shared planning scheme and 50% with an individual local planning (the order in which the trucking companies and their fleet are added to the SPS is randomised; no prior assumptions are made about which companies will join the system first). It is clear that by doing no planning or optimisation, i.e., random planning, the traffic times are much greater than in the case of local and global optimisation, which is to be expected. As expected, one can see an improvement in the distribution of traffic times in the case of global optimisation when compared to the distribution obtained with local optimisation. On average the reduction in traffic times amounts to $13.2 \pm 1.3\%$ (errors denote standard deviation across different simulations) when going from local optimisation to global optimisation. When 50% of the trucks and their corresponding companies have joined the SPS, the average time spent in traffic is already reduced by $9.2 \pm 1.3\%$ when compared to the case where all companies apply an individual real-time planning. These results are summarised in the first column of Table 3.

Figure 5 Distribution of time spent in traffic per truck for the three different planning strategies: random, local and global (see online version for colours)



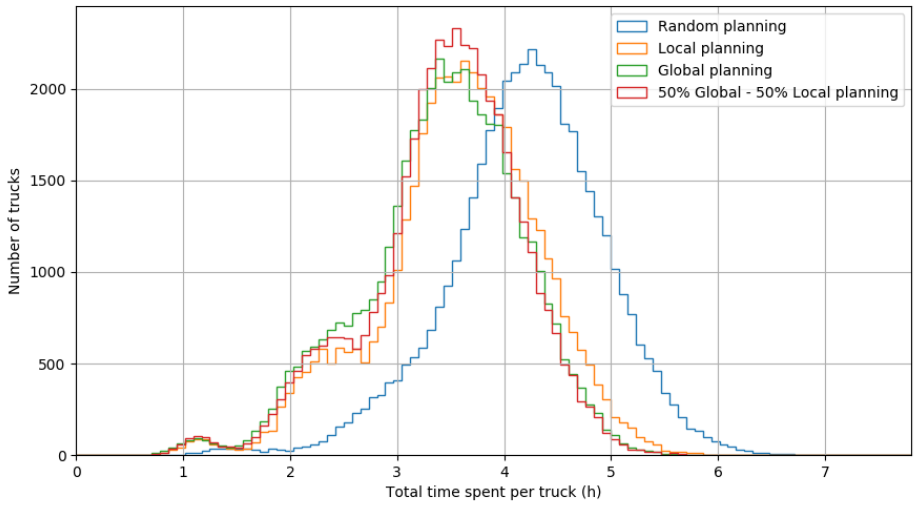
In Figure 6, the distribution of the total time spent in the simulation by each truck is given for the different strategies, this includes traffic time, time spent in the terminals and the time spent when loading or unloading a container at its origin or destination. Again, when looking at the distributions, one can see a clear difference when comparing the random strategy to the other two. Again there is an improvement when going from individual planning to a shared planning: the average total time spent per truck is decreased by $4.4 \pm 0.8\%$. In the transitional case where only 50% of the trucks use the SPS, the average total time per truck (which includes the 50% of trucks that are not

part of the SPS) is already reduced by $3.8 \pm 0.7\%$. The fact that these differences are smaller compared to the difference in traffic times has two underlying causes. The first reason is that the relative differences are smaller since they are comparing the total time, of which the traffic times are only a part, and this is the factor on which is focused in the optimisation schemes. The second reason is that due to the better routing and traffic times in the global optimisation scheme as compared to the local optimisation scheme, trucks arrive slightly earlier in general and all orders are finished in a shorter time-period. Therefore, the times at which trucks arrive at terminals are slightly less spread out in time (since the load at the terminals is not explicitly taken into account in the optimisation), resulting in slightly longer queues at terminals and thus greater total times. However, the total effect of global optimisation is still positive and substantial.

Table 3 Average time spent in traffic (s), average total time spent (s), average total distance driven per truck (km), and the average amount of empty kilometres driven for the three planning strategies

Strategy	*Traffic time*	*Total time*	*Distance*	*Empty kms*
Random planning	9,671	15,126	124.1	57.8
Local optimisation	7,022	12,856	92.0	26.7
Global optimisation	6,094	12,290	82.5	17.9
50% global/50% local	6,373	12,373	87.2	22.5

Figure 6 Distribution of total time (in traffic, terminals and while loading) spent per truck for the three different planning strategies: random, local and global (see online version for colours)



In the third column of Table 3, the average total distance driven per truck is displayed. This can readily be converted to vehicle-km, knowing that there are 4586 trucks active in the simulation (the unit vehicle-km is defined as the product of the number of vehicles on a given road or traffic network and the average length of their trips measured in kilometres). One then finds that by applying the locally optimising planning as opposed

to the random planning, there is a reduction of $25.9 \pm 0.4\%$ in vehicle-km. Comparing the local optimisation to the globally optimising SPS, the reduction amounts to $10.3 \pm 0.3\%$ vehicle-km. When only 50% of trucks are part of the SPS and the rest applies an individual planning, the reduction is $5.1 \pm 0.4\%$ vehicle-km. The introduction of a real-time SPS thus leads to a substantial reduction in vehicle-km. Note that the relative reduction in transit times compared to the reduction in vehicle-km is slightly greater. This is a direct consequence of reduced congestion on the traffic network due to more efficient routing. Comparing the case of individual planning to the shared planning, the average velocity of trucks is increased by 3.2%. The last column of Table 3 contains the average empty kilometres driven per truck. The amount of empty kilometres driven is reduced by 33.0% when going from an individual planning to a collaborative system.

6.2 *Sensitivity: increasing the number of orders*

Now the different planning strategies will be tested and compared with an increased number of orders. Suppose for example that we underestimated the number of orders or that there is an increase in orders in the future. To this end the number of orders will be increased by 30%, all other parameters will be kept the same as in the previous subsection.

Figure 7 Distribution of time spent in traffic per truck for the three different planning strategies: random, local and global; with the number of orders increased by 30% (see online version for colours)

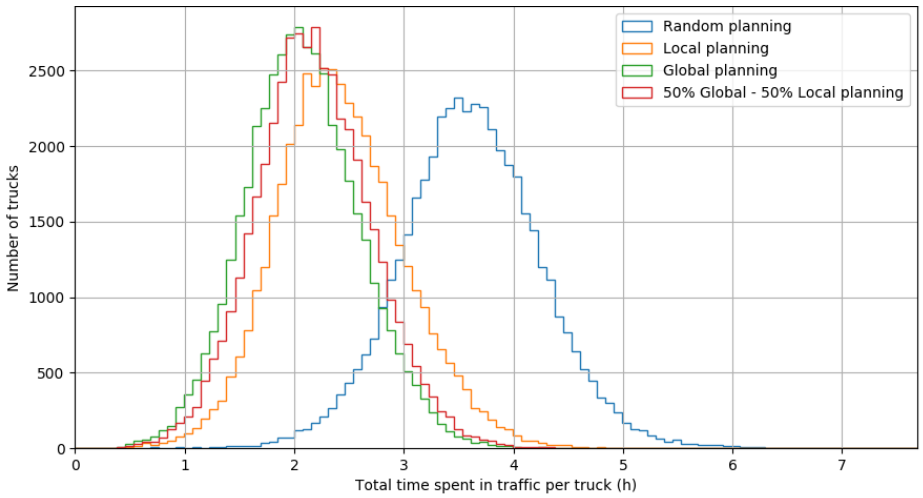


Figure 7 shows the distribution of the times spent in traffic during the simulation per truck. The results are similar to the previous ones. When going from individual planning to the SPS, one can again see a clear improvement, amounting to a reduction of $14.4 \pm 4.0\%$ in time spent in traffic. In the transitional case, the traffic times are on average reduced by $10.1 \pm 1.5\%$. In Figure 8, the distribution of the total times spent in the simulation per truck is shown. Going from individual planning to the SPS, the reduction in average total time spent is $5.4 \pm 1.8\%$; again smaller than the reduction in

time spent in traffic, but still substantial. The difference between a complete individual planning and the transitional case is $4.4 \pm 0.8\%$. The average times and distances for each case are summarised in Table 4. We thus find that the improvements found are consistent with the results found earlier. This is a promising result as we probably underestimated the total number of orders by assuming they all consist of 40-foot-long containers. Also, the total throughput in the port is increasing every year, meaning the number of orders today will already be greater.

Figure 8 Distribution of total time (in traffic, terminals and while loading) spent per truck for the three different planning strategies: random, local and global; with the number of orders increased by 30% (see online version for colours)

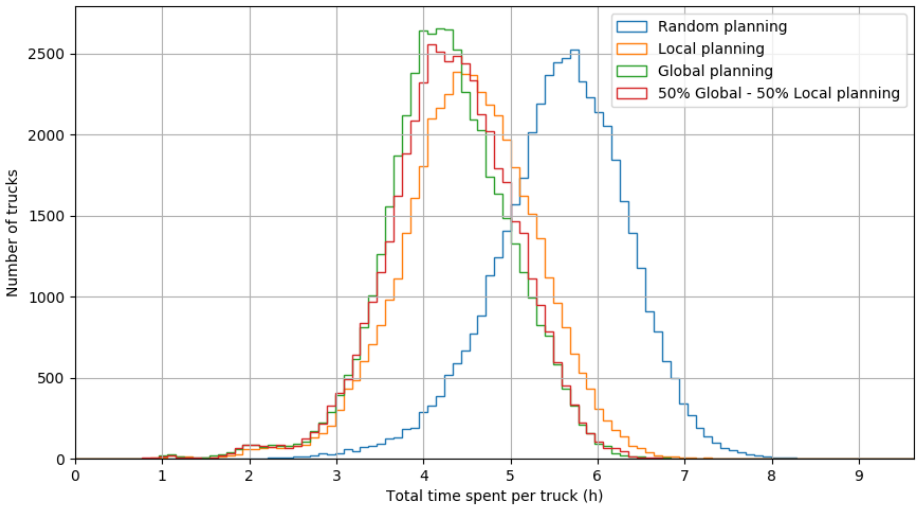


Table 4 Average time spent in traffic (s), average total time spent (s) and average total distance driven per truck (km) for the three planning strategies with the number of orders increased by 30%

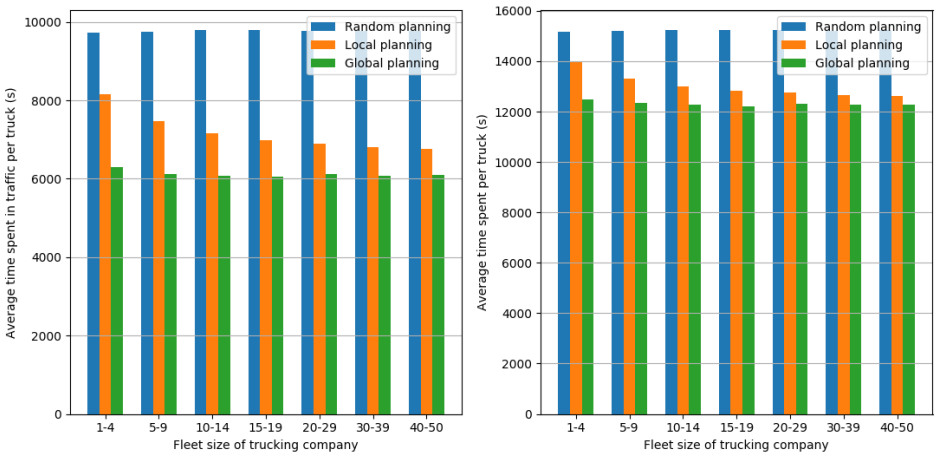
Strategy	*Traffic time*	*Total time*	*Distance*
Random planning	13,008	20,102	162.0
Local optimisation	8,717	16,265	115.6
Global optimisation	7,465	15,386	102.5
50% global/50% local	7,839	15,554	108.7

6.3 Influence of fleet size

Let us now look in more detail at the impact of the fleet size of a trucking company on its improvements when joining the SPS. In Figure 9, the average traffic time per truck for companies with different fleet sizes [Figure 9(a)] and the average total time per truck for different company sizes [Figure 9(b)] are depicted. The results are summarised in Tables 5 and 6. From this, it can be seen that joining a coalition in the shared planning is relatively more beneficial for trucking companies with smaller fleets. This is a result which one intuitively expects, as the pool of potential orders to choose from is increased

more for smaller companies joining the SPS than for larger companies. In order to make the coalition more stable and the division of profit more equal, side payments could be introduced, this is however outside the scope of our work.

Figure 9 (a) The average time spent in traffic per truck (b) Total time for different fleet sizes of the trucking companies under different planning strategies (see online version for colours)



(a) (b)

Table 5 Results for average traffic times per truck for companies with different fleet sizes

Fleet size	*1–4*	*5–9*	*10–14*	*15–19*	*20–29*	*30–39*	*40–50*
Random planning (s)	9,734	9,742	9,782	9,807	9,777	9,760	9,780
Local optimisation (s)	8,157	7,473	7,149	6,976	6,903	6,810	6,760
Global optimisation (s)	6,288	6,127	6,086	6,043	6,113	6,074	6,088
Gain: random → local (%)	16.2	23.3	26.9	28.8	29.4	30.2	30.9
Gain: local → global (%)	22.9	18.0	14.9	13.4	11.4	10.8	9.9

Table 6 Results for average total time spent per truck for companies with different fleet sizes

Fleet size	*1–4*	*5–9*	*10–14*	*15–19*	*20–29*	*30–39*	*40–50*
Random planning (s)	15,185	15,194	15,240	15,252	15,227	15,209	15,224
Local optimisation (s)	13,956	13,313	12,993	12,812	12,746	12,653	12,608
Global optimisation (s)	12,477	12,330	12,277	12,223	12,297	12,263	12,276
Gain: random → local (%)	8.1	12.4	14.7	16.0	16.3	16.8	17.2
Gain: local → global (%)	10.6	7.4	5.5	4.6	3.5	3.1	2.6

7 Discussion and conclusions

In this work, a shared real-time on-the-go planning system for road carriers in a port is presented. This SPS was validated in a detailed large-scale simulation based on the case

of the Port of Antwerp. It was illustrated that by introducing a shared planning leads to considerable improvements compared to an individualised planning. In the case of the simulation of drayage in the Port of Antwerp:

- travel times are on average reduced by 13.2%
- a reduction of 10.3% in vehicle-km
- a reduction of 33.0% in empty kilometres
- an increase in average speed in traffic of 3.2%
- a reduction of 4.4% in total time (including time spent at terminals, during loading and unloading of containers, ...), or an increase of 4.4% in the number of orders that can be handled.

This increase in efficiency is not only positive for the participating road carriers, but may benefit the whole supply chain by decreasing road congestion around the port, reducing carbon emissions, and transportation costs, and increasing the system-wide truck capacity.

In the description and experiments in this work, the sharing of orders was kept in balance on the scale of one day. In principle, it is also possible to extend this to longer time scales (i.e., building up credit over a longer period of time). It was also demonstrated that when 50% of the trucks and their corresponding companies (picked at random) join the shared system, the improvements in efficiency are already considerable. This illustrates that it is not necessary for all companies to join the system before benefits are noticeable, which creates an incentive for trucking companies to join the SPS in the early phase in practice.

Apart from being able to incorporate real-time information in planning, another major advantage of the real-time planning is that it allows for great flexibility in terms of trucks and orders joining or leaving the system. Concerning the second major group of stakeholders in this story, container terminals can also benefit from this system. Firstly, greater efficiency in container transport on the side of trucks and trucking companies means that more orders can be processed each day. Secondly, this SPS would allow for great transparency towards the terminals. For all trucks using this planning system, they can get precise information on when to expect which truck and for which order; this information could be used to further optimise their internal operations.

The presented research has several limitations, which are potential directions for future research. A first point is the simplified simulation of the container terminals. Further research could entail a more detailed simulation of the internal operations of each individual terminal and the resulting truck turnaround times. Moreover, the optimisation of the shared planning could then also explicitly take into account the current load and expected future load on each of the container terminals. Related to this point, a second limitation of the presented work is that the incorporation of a shared planning with a TAS was not yet worked out in detail. Collaboration within a TAS is an interesting case on its own and could open up other opportunities. Finally, the improvement in using an SPS was also studied for individual trucking companies as a function of their respective fleet size. As could be expected, it was found that the relative gain is higher for smaller trucking companies compared to ones with a larger fleet. The issue of the correct assignment of side payments between trucking companies in order to make the coalition stable has not been covered in this work. As mentioned

in the literature review, efficient methods for determining profit allocations in large coalitions in the context of collaboration in transport remains an open problem and a possible subject for future research.

Acknowledgements

This work was made possible by the MobiliData-TruckGuidance, COOCK, AI4FL (imec-icon) and OptiRouts (imec-icon) projects, and the UGent grand BOF/STA/202009/039.

References

Agarwal, R., Ergun, Ö., Houghtalen, L. and Ozener, O.O. (2009) 'Collaboration in cargo transportation', *Optimization and Logistics Challenges in the Enterprise*, pp.373–409, Springer.

Akamatsu, T. (2001) 'An efficient algorithm for dynamic traffic equilibrium assignment with queues', *Transportation Science*, Vol. 35, No. 4, pp.389–404.

Bast, H., Delling, D., Goldberg, A., Müller-Hannemann, M., Pajor, T., Sanders, P., Wagner, D. and Werneck, R.F. (2016) 'Route planning in transportation networks', *Algorithm Engineering, Lecture Notes in Computer Science (LNTCS)*, Springer, Vol. 9220, pp.19–80.

Bellman, R. (1958) 'On a routing problem', *Quarterly of Applied Mathematics*, Vol. 16, No. 1, pp.87–90.

Brockfeld, E., Kühne, R.D. and Wagner, P. (2004) 'Calibration and validation of microscopic traffic flow models', *Transportation Research Record*, Vol. 1876, No. 1, pp.62–70.

Caballini, C., Sacone, S. and Saeednia, M. (2016) 'Cooperation among truck carriers in seaport containerized transportation', *Transportation Research Part E: Logistics and Transportation Review*, Vol. 93, No. 1, pp.38–56.

Carlan, V., Naudts, D., Audenaert, P., Lannoo, B. and Vanelslander, T. (2019) 'Toward implementing a fully automated truck guidance system at a seaport: identifying the roles, costs and benefits of logistics stakeholders', *Journal of Shipping and Trade*, Vol. 4, No. 1, pp.1–24.

Chan, F.T. and Zhang, T. (2011) 'The impact of collaborative transportation management on supply chain performance: a simulation approach', *Expert Systems with Applications*, Vol. 38, No. 3, pp.2319–2329.

Charypar, D., Axhausen, K.W. and Nagel, K. (2007) 'Event-driven queue-based traffic flow microsimulation', *Transportation Research Record*, Vol. 2003, No. 1, pp.35–40.

Dai, B. and Chen, H. (2011) 'A multi-agent and auction-based framework and approach for carrier collaboration', *Logistics Research*, Vol. 3, No. 2, pp.101–120.

Di Francesco, M. and Rosini, M. (2014) *Rigorous Derivation of the Lighthill-Whitham-Richards Model from the Follow-the-Leader Model as Many Particle Limit*, arXiv preprint arXiv:1404.7062.

Do, T.M.H., Park, G-K., Choi, K-H. and Yang, X. (2021) 'The application of uncertain three-player two-stage game to the competition among shipping alliances', *International Journal of Shipping and Transport Logistics*, Vol. 13, No. 6, pp.600–623.

Eisenblätter, B., Santen, L., Schadschneider, A. and Schreckenberg, M. (1998) 'Jamming transition in a cellular automaton model for traffic flow', *Physical Review E*, Vol. 57, No. 2, p.1309.

Eissfeldt, N.G. (2004) *Vehicle-Based Modelling of Traffic. Theory and Application to Environmental Impact Modelling*, PhD thesis, Universität zu Köln.

FOD Mobiliteit (2017) *Editie 2017*, FOD Mobiliteit [online] https://mobilit.belgium.be/nl/mobiliteit/woon_werkverkeer/editie_2017 (accessed 21 March 2018).

Ferrara, A., Sacone, S. and Siri, S. (2018) 'Microscopic and mesoscopic traffic models', *Freeway Traffic Modelling and Control*, pp.113–143, Springer.

Fredman, M.L. and Tarjan, R.E. (1987) 'Fibonacci heaps and their uses in improved network optimization algorithms', *Journal of the ACM (JACM)*, Vol. 34, No. 3, pp.596–615.

Gawron, C. (1998a) 'An iterative algorithm to determine the dynamic user equilibrium in a traffic simulation model', *International Journal of Modern Physics C*, Vol. 9, No. 3, pp.393–407.

Gawron, C. (1998b) *Simulation-Based Traffic Assignment*, PhD thesis, Universität zu Köln.

Geisberger, R., Sanders, P., Schultes, D. and Vetter, C. (2012) 'Exact routing in large road networks using contraction hierarchies', *Transportation Science*, Vol. 46, No. 3, pp.388–404.

Gipps, P.G. (1981) 'A behavioural car-following model for computer simulation', *Transportation Research Part B: Methodological*, Vol. 15, No. 2, pp.105–111.

Göthe-Lundgren, M., Jörnsten, K. and Värbrand, P. (1996) 'On the nucleolus of the basic vehicle routing game', *Mathematical Programming*, Vol. 72, No. 1, pp.83–100.

Gracia, M.D., Mar-Ortiz, J. and González-Ramírez, R.G. (2019) 'The impact of operational strategies on vessel handling times: a simulation approach', *International Journal of Shipping and Transport Logistics*, Vol. 11, No. 4, pp.287–315.

Huynh, N., Smith, D. and Harder, F. (2016) 'Truck appointment systems: where we are and where to go from here', *Transportation Research Record*, Vol. 2548, No. 1, pp.1–9.

Islam, S. and Olsen, T. (2014) 'Truck-sharing challenges for hinterland trucking companies: a case of the empty container truck trips problem', *Business Process Management Journal*, Vol. 20, No. 2, pp.290–334.

Krajewska, M.A., Kopfer, H., Laporte, G., Ropke, S. and Zaccour, G. (2008) 'Horizontal cooperation among freight carriers: request allocation and profit sharing', *Journal of the Operational Research Society*, Vol. 59, No. 11, pp.1483–1491.

Lehe, L. (2014) [online] https://setosa.io/blog/2014/09/02/gridlock/ (accessed 1 July 2021).

Lopez, P.A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y-P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P. and Wießner, E. (2018) 'Microscopic traffic simulation using SUMO', *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, pp.2575–2582.

Notteboom, T. (2009) *The Relationship between Seaports and the Intermodal Hinterland in Light of Global Supply Chains: European Challenges*, OECD.

OpenStreetMap Contributors (2020) *Planet Dump* [online] https://www.openstreetmap.org (accessed 7 May 2020).

Papageorgiou, M. (1998) 'Some remarks on macroscopic traffic flow modelling', *Transportation Research Part A: Policy and Practice*, Vol. 32, No. 5, pp.323–329.

Port of Antwerp (2020) *Port of Antwerp Facts and Figures 2020*.

Ramshaw, L. and Tarjan, R.E. (2012) *On Minimum-Cost Assignments in Unbalanced Bipartite Graphs*, Tech. Rep. HPL-2012-40R1, HP Labs, Palo Alto, CA, USA.

Richards, P.I. (1956) 'Shock waves on the highway', *Operations Research*, Vol. 4, No. 1, pp.42–51.

Rubbrecht, I. (2022) *Economic Importance of the Belgian Maritime and Inland Ports – Report 2020*, Technical report, NBB Working Paper.

Saprykin, A., Chokani, N. and Abhari, R.S. (2020) 'Gridlock resolution in a GPU-accelerated traffic queue model', *Procedia Computer Science*, Vol. 170, No. 1, pp.681–687.

Schulte, F., Lalla-Ruiz, E., González-Ramírez, R.G. and Voß, S. (2017) 'Reducing port-related empty truck emissions: a mathematical approach for truck appointments with collaboration', *Transportation Research Part E: Logistics and Transportation Review*, Vol. 105, No. 1, pp.195–212.

Seibold, B., Flynn, M.R., Kasimov, A.R. and Rosales, R.R. (2012) *Constructing Set-Valued Fundamental Diagrams from Jamiton Solutions in Second Order Traffic Models*, arXiv preprint arXiv:1204.5510.

Shao, Q., Huang, M., Zhang, S. and Zhang, Y. (2022) 'Simulation of truck arrivals at container terminal based on the interactive truck appointment system', *International Journal of Shipping and Transport Logistics*, Vol. 14, Nos. 1–2, pp.141–171.

Simatupang, T.M. and Sridharan, R. (2005) 'The collaboration index: a measure for supply chain collaboration', *International Journal of Physical Distribution & Logistics Management*, Vol. 35, No. 1, pp.44–62.

Sirimanne, S.N., Hoffman, J., Juan, W., Asariotis, R., Assaf, M., Ayala, G., Benamara, H., Chantrel, D., Hoffmann, J., Premti, A. et al. (2019) *Review of Maritime Transport 2019*, United Nations Conference on Trade and Development, Geneva, Switzerland.

Sniedovich, M. (2006) 'Dijkstra's algorithm revisited: the dynamic programming connexion', *Control and Cybernetics*, Vol. 35, No. 3, pp.599–620.

Treiber, M., Hennecke, A. and Helbing, D. (2000) 'Congested traffic states in empirical observations and microscopic simulations', *Physical Review E*, Vol. 62, No. 2, p.1805.

Yang, Q. and Koutsopoulos, H.N. (1996) 'A microscopic traffic simulator for evaluation of dynamic traffic management systems', *Transportation Research Part C: Emerging Technologies*, Vol. 4, No. 3, pp.113–129.

Zhou, X. and Taylor, J. (2014) 'Dtalite: a queue-based mesoscopic traffic simulator for fast model evaluation and calibration', *Cogent Engineering*, Vol. 1, No. 1, p.961345.

Appendix

The traffic network

The network considered in this simulation is the complete road network in a rectangle of about 30 km $\times$ 25 km around Antwerp and its port. The networks consists of $|V| = 20{,}489$ nodes or intersections and $|A| = 46{,}685$ arcs or roads connecting them (only roads where trucks are allowed are included). The traffic network data was obtained from OpenStreetMap (2020) and was cleaned up: arcs were filtered to only include roads that allow trucks, isolated parts of the network were removed, the maximum speed limits were set to 90 km/h, redundant nodes were contracted, the curvature of the road was discarded in the simulation and only used for visualisation purposes, etc.

A mesoscopic traffic model

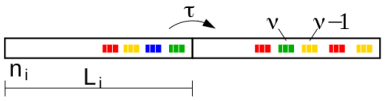
The framework that is used in this study is the one of mesoscopic traffic modelling, more specifically based on (state-dependent) queueing theory. In terms of queueing theory, each link of a street network is regarded as a queue, obeying the FIFO (first in, first out) principle, i.e., a service device operating at a certain service rate which corresponds to the flow capacity of the link, being the maximum throughput in [vehicles/h] which can be maintained. Queues of vehicles (congestion) occur in the system, whenever the current demand exceeds the flow capacity of a service. In consequence, vehicles queue up in front of the service device, and experience additional waiting times before being served. The total time a vehicle spends on a link, therefore,

equals the sum of the waiting time due to congestion and the service time. Moreover, the service times will depend on the state (i.e., the density) of the considered link. This allows to replicate the phase transition that occurs in real vehicular traffic systems, namely from the free flow phase to the jamming phase, where a jam or *shock wave* propagates backwards through the system.

The model that will be presented here is, as was mentioned in the main article, based on the work in Eissfeldt (2004), the μ-queue model, although some additions and adaptations are made. In this model, roads are divided into segments, each segment being a separate queue. The length of these segments should be chosen long enough, such that enough vehicles fit into it. There is not really an upper limit on the lengths of the segments, when one wants a more fine-grained study of certain roads, one can divide it into more segments, though the general dynamics do not depend on it. In this study, high resolutions for dynamics on roads are not necessary, as long as the overall travel times are realistic. Also keeping in mind computation efficiency, the segments are chosen to be the links between intersections (note that intersections also include points where there is a change in a property of the road, such as the speed limit or the number of lanes).

Each road segment is thus represented as a queue with a *storage* capacity $c_i = \lfloor \frac{L_i}{L_{\text{vehicle}}} \rfloor$ with L_i the length of the road segment and L_{vehicle} the length of a vehicle, see Figure 10.

Figure 10 Two adjacent road segments/queues (see online version for colours)



Note: n_i is the number of vehicles on segment i.

Source: Eissfeldt (2004)

When a vehicle ν enters a segment i at time $t^{\nu}_{i,en}$ it gets assigned a minimum exit time of

$$t^{\nu}_{\text{min}} = t^{\nu}_{i,en} + \frac{L_i}{v_i} \tag{4}$$

where v_i is the speed limit on segment i. When time t^{ν}_{min} has elapsed, the vehicle can move to the next segment on its route, provided that the next segment j is not full, $n_j < c_j$ (where n_j is the number of vehicles currently on segment j), and that the flow capacity constraint at the end of the current segment is obeyed. The flow capacity constraint ensures the flow q does not exceed the maximal possible flow Q_i at a certain road segment i. This can be expressed in terms of a time-headway between two vehicles τ_i, which can be seen as an additional waiting time

$$Q_i = \frac{1}{\tau_i} \tag{5}$$

Vehicle ν is not allowed to exit segment i before time $t^{\nu-1}_{i,ex} + \tau_i$, where $t^{\nu-1}_{i,ex}$ denotes the exit time of the last vehicle leaving segment i. Additionally there is a flow capacity

constraint on the next segment j which ν is about to enter, τ_j, such that vehicle ν can enter segment j only when time $t_{j,en}^{\nu-1} + \tau_j$ has elapsed, where $t_{j,en}^{\nu-1}$ denotes the time when the last vehicle entered segment j. Both τ_i and τ_j depend on the state/density in segments i and j, let us therefore denote both τ_i and τ_j by one symbol τ_{ij}.

Thus vehicle ν is allowed to exit its current segment i and enter its following segment j when it is at the front of the queue and when the following conditions are met

$$\begin{cases} n_j < c_j \\ t \geq \max\{t_{i,en}^{\nu} + \frac{L_i}{v_i}, t_{i,ex}^{\nu-1} + \tau_{ij}, t_{j,en}^{\nu-1} + \tau_{ij}\} \end{cases} \tag{6}$$

It is the second and third argument of $\max\{\cdot,\cdot,\cdot\}$ which introduce the interaction between vehicles in the model. In order to incorporate the dynamics of traffic jam formation and its backward propagation, τ_{ij} has to depend on the state of segments i and j. To this end, the following form is proposed (based on the μ_3-Queue model from Eissfeldt, 2004)

$$\tau_{ij} = \begin{cases} \tau_{\text{ff}} & \text{if } n_i < n_i^{jam} \\ \tau_{\text{jf}} & \text{if } n_i \geq n_i^{jam} \text{ and } n_j < n_j^{jam} \\ \tau_{\text{jj}} n_j + c_j(\tau_{\text{jf}} - \tau_{\text{jj}}) & \text{if } n_i \geq n_i^{jam} \text{ and } n_j \geq n_j^{jam} \end{cases} \tag{7}$$

where $n_i^{jam} = \lfloor \rho_{jam} L_i \rfloor$ is the occupation at which the phase transition from free flow to jammed traffic occurs, and τ_{ff} and τ_{jf} are tunable parameters of the model. Note that the first case covers both the case where $n_j < n_j^{jam}$ and $n_j \geq n_j^{jam}$. The third parameter τ_{jj} is chosen such that τ_{ij} is continuous at $n_j = n_j^{jam}$:

$$\tau_{\text{jj}} = \frac{c_j - 1}{c_j - n_j^{jam}} \tau_{\text{jf}} \tag{8}$$

The vehicles that will be modelled here are trucks (including a trailer), which are assumed to have a standard length of $L_{\text{vehicle}} = 18$ m and a maximum speed of 90 km/h = 25 m/s. The other parameters are set to $\tau_{\text{ff}} = 2.5$ s and $\tau_{\text{jf}} = 3.5$ s, which are based on Eissfeldt (2004) but were rescaled from cars to trucks. Note that the assumption that these capacity constraints τ_{xx} which generate congestion are independent of the free flow velocity of the segment is an approximation and a limitation of the model.

The intuition behind $\tau_{ij} \propto n_j$ in the case of jamming is the following. Imagine a vehicle ν at the front of a road segment i which wants to move to a full segment j. Now the vehicle μ at the front of segment j leaves this segment at a certain time. The free site or hole generated in this way needs a certain time to reach the upstream segment, proportional to the number of vehicles left on this segment n_j. The reason for this is that people/vehicles are unable to move coordinated at once, each vehicle reacts with a certain delay to the movement of the vehicle directly in front of it (this is the deep reason behind the formation of traffic jams). Vehicle ν that wants to enter this segment j has to wait until the hole reaches the upstream side of this segment and thus $\tau_{ij} \propto n_j$.

The jamming density ρ_{jam} and thus the threshold n_i^{jam} depends on the maximum allowed speed on that segment, the higher the velocity of the traffic, the lower ρ_{jam}. The intuition behind this is that the faster vehicles are moving, the more distance is

needed between them to have enough time to react to sudden movements of the vehicles in front (again, the slow reaction of humans). The exact relation is given by (see, e.g., Eisenblätter et al., 1998)

$$\rho_{jam}(v) = \frac{\rho_{\max}}{1 + \frac{(\rho_{\max} - \rho^0_{jam})v}{\rho^0_{jam} v_0}} \tag{9}$$

where v is the speed limit on that road and (v_0, ρ^0_{jam}) is a reference point, empirical parameters, which are chosen to be $(13.89 \text{ m/s}, 1/42 \text{ m}^{-1})$, for trucks of length ~ 18 m. These values are based on fundamental diagrams for cars such as in Seibold et al. (2012) and rescaled to trucks. This is a slight adaptation made to the model in Eissfeldt (2004), where the density at which the transition occurs was assumed to be constant.

The dynamics governed by equations (6) and (7) reproduce the well known fundamental diagrams of traffic flow, and are able to model traffic jams that propagate backwards with a certain finite velocity. Some additions have to be carried out. Firstly, road/segment storage capacities c_i as well as flow capacities τ have to be rescaled with the varying number of lanes l_i on a certain road segment i.

$$\begin{cases} c_i = \left\lfloor \frac{L_i}{L_{\text{vehicle}}} \right\rfloor & \rightarrow \quad c_i = l_i \left\lfloor \frac{L_i}{L_{\text{vehicle}}} \right\rfloor \\ n_i^{jam} = \lfloor \rho_{jam} L_i \rfloor & \rightarrow \quad n_i^{jam} = l_i \lfloor \rho_{jam} L_i \rfloor \\ \tau_{xx} & \rightarrow \quad \tau_{xx}/l_i \end{cases} \tag{10}$$

Secondly, junctions are modelled in a relatively elementary way, they get assigned a certain flow capacity constraint as well. Until now, only flow constraints on the arcs/roads of the traffic network separately were considered, but at junctions or crossings, outgoing and ingoing traffic can mix between different roads. In order to keep things realistic, and not let vehicles move 'through' one another, the junctions themselves need to obey a flow constraint, which is at most the maximal free flow of the adjacent road with the highest flow capacity, i.e., $\tau_{junc} = \tau_{\text{ff}} / \max_i(l_i)$. The complete set of conditions that have to be fulfilled in order for a vehicle to transit from a segment i to the next segment j on its route is given by

$$\begin{cases} n_j < c_j \\ t \geq \max \left\{ t^{\nu}_{i,en} + \frac{L_i}{v_i}, t^{\nu-1}_{i,ex} + \frac{\tau_{ij}}{l_i}, t^{\nu-1}_{j,en} + \frac{\tau_{ij}}{l_j}, t^{\nu-1}_{junc} + \frac{\tau_{\text{ff}}}{\max_i(l_i)} \right\} \end{cases} \tag{11}$$

where $t^{\nu-1}_{junc}$ denotes the time when the last vehicle crossed the junction to which i and j are adjacent.

A third addition that is made is that the jamming density of an off-ramp is set equal to the maximum occupation, $n_i^{jam} = c_i$. This is done in order to suppress the influence that an off-ramp has on the flow of the corresponding highway (in Eissfeldt, 2004 this is done by introducing extra 'storage segments', which function in a similar way).

The model described so far allows us to put vehicles in the traffic network with a certain predetermined route and let them drive through the network and interact with one another. For the time-evolution, the simulation is updated in an event-driven way. Instead of using time steps explicitly, the temporal process is modelled as a

sequence of events which take place at real-valued points in time, given by equation (11). The leaving process of a vehicle can be associated with an event. Because of FIFO conditions, only the vehicles at the downstream ends of the segments, and for which their next segment is not completely filled, have to be considered. They can be stored efficiently using a priority queue (implemented by a heap) which gives back the earliest next event. The simulation is updated by extracting the vehicle with the earliest moving time and updating the simulation time to that time. When a vehicle ν transfers from the head of its queue to the next segment on its route, the vehicle behind it, $\nu + 1$, is inserted in the priority queue (on the condition that the segment it will move to is not full). Also, when a vehicle leaves a full queue, it is again possible to move to this segment, so vehicles at the heads of the segment on the back of this updated segment are added to the priority queue. This event-driven updating allows the simulation to run efficiently and simulate ten thousands of vehicles in a large network with hundreds of thousands of arcs/roads.

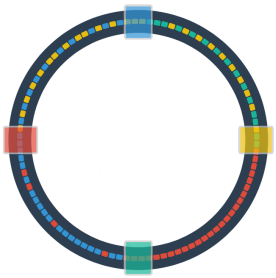
In the type of traffic model described so far, it is possible for cyclic gridlocks to occur when the load on the traffic network is increased. In the next section, we will discus the methodology with which this is prevented; amounting to another addition made to the version by Eissfeldt (2004).

Each model has its limitations and so does this one. As we already mentioned, a first limitation is the fact that the capacity constraints τ_{xx} are independent of the free flow velocity of the segment is an approximation. Secondly, the intersections in the traffic model are modelled in a simplified way, e.g., traffic lights are not taken into account. Thirdly, the presented model only handles homogeneous traffic, i.e., all vehicles are of the same size, which is not a problem for our work since we only consider truck traffic.

Depth-first-search for gridlock detection in a queueing network

When the load on the traffic network increases, it is possible gridlocks occur on closed loops in the network. This happens when a loop of full arcs/segments is formed, see for example Figure 11: each vehicle on the front of the queue wants to continue its journey to it end destination (the color of the vehicle), but is unable to because the next segment is full.

Figure 11 An example of a cyclic gridlock (see online version for colours)



Source: Lehe (2014)

Gridlocks are rare situations which can occur in real-life traffic situations, but the presented model is a bit too sensitive to these particular ones, especially for small loops. In simulations, this problem is often avoided by temporarily increasing the storage capacities (Saprykin et al., 2020), setting them to infinity (*point queues*) (Zhou and Taylor, 2014) or pushing the vehicle to next queue even if it is full (Charypar et al., 2007). Here the issue is resolved in a different way; an extra condition is added to equation (11) for a vehicle to be allowed to move and enter the priority queue. Apart from the condition that the next segment has to have a free spot, it can not be part of a *critical cycle*. By critical cycle, we mean that if the next segment j has only one free spot left, $n_j = c_j - 1$, and it forms a cycle with otherwise completely filled segments, then j is part of a critical cycle. This rule ensures that every cycle in the traffic network has at least 1 free spot, such that the traffic in this cycle can keep on moving. The intuition behind this is that on for example a roundabout, there is always some free space, such that the vehicles can keep on moving.

Now, in order to maintain the property that no full cycles exist in the traffic network, the extra condition added to equation (11) for a move to be allowed and thus a vehicle to be added to the priority queue is (again considering a move from segment i to segment j):

$$C_j \setminus C_i = \varnothing \tag{12}$$

where C_i denotes the set of critical cycles that contain segment i. This condition allows critical cycles to evolve and dissolve, whilst still preventing the formation of full cycles and thus cyclic gridlocks. When the move from i to j has taken place, the cycles in the set $C_i \setminus C_j$ can be cleared, for example, when the vehicle moves to an arc j that is not part of a critical cycle, the cycles on i can be dissolved.

In order to detect these critical cycles, when a vehicle is to be put in the priority queue of vehicles that want to move, its next segment j is checked for such a cycle with an algorithm based on depth-first-search (DFS), with some modifications. The first difference is that, instead of exploring nodes, it explores arcs. Secondly, arcs are not marked as visited and can be visited more than once. The goal of this adapted version of DFS will be to find cycles of completely filled arcs and exactly 1 arc with one free spot (*quasi-full*). The algorithm will thus only explore full arcs and, if no quasi-full arc was encountered so far, quasi-full arcs. The pseudocode is given below in Algorithm 4; here the term *outgoing* arc a_j from arc a_i is used for an arc a_j that has his starting point at the endpoint of the other arc a_i. The algorithm will find all critical cycles containing the starting arc a_0. As the code is given, it is also possible that it finds cycles of arcs that are all completely full, however, such cycles cannot form in the traffic simulation since this is explicitly prevented with condition (12).

For realistic situations, this can be done quite efficiently, as the DFS only has to explore segments that are fully occupied and 1 that is nearly fully occupied (one free spot left). The subnetwork of these completely filled arcs remains small in practice, such that cycles can be found quickly, or one can quickly stop searching. Every critical cycle found is saved, as they often persist for some time and it would create overhead to search for them over and over again. All this allows one to carry out this check with little extra overhead.

Algorithm 4 DFS-Critical-Cycle(G, a_0)

```
1: S_a = empty stack for arcs
2: S_p = empty stack for paths, a path being a set of arcs
3: S_b = empty stack for booleans, indicating if a quasi-full arcs was visited
4: S_a.push(a_0)
5: S_p.push({a_0})
6: S_b.push(a_0.isQuasiFull())
7: while S_a not empty do
8:     a ← S_a.pop()
9:     p ← S_p.pop()
10:    b ← S_b.pop()
11:    if a has a_0 as an outgoing arc then
12:        save new Cycle(p)
13:        continue
14:    for arcs o outgoing from a do
15:        if (o.isFull() or (o.isQuasiFull() and not b)) and not p.contains(o) then
16:            S_a.push(0)
17:            S_p.push(path(p ∪ {o}))
18:            S_b.push(b or o.isQuasiFull())
```

Routing

In order to find shortest paths in networks, and in particular in road networks, many algorithms have been constructed and studied, such as Dijkstra's algorithm (Sniedovich, 2006), the Bellman-Ford algorithm (Bellman, 1958), contraction hierarchies (Geisberger et al., 2012), and many more, for an overview see Bast et al. (2016). Many algorithms that have been proposed are based on Dijkstra's algorithm and each of them offers a certain trade-off between execution time (at time of request) and memory usage/precomputation time. The routing algorithm used here is based on Dijkstra's algorithm, where certain search trees will be kept in memory for efficient querying during the simulation. The specific way in which the Dijkstra algorithm is implemented has a great impact on the computational efficiency. Here, the algorithm will make use of a priority queue implemented by a binary heap, resulting in a worst-case time-performance of $\Theta((|A| + |V|) \log |V|)$ for a request of the shortest path between two random nodes s and t. As will be seen later, many requests will have either the starting node s or target node t on one of five nodes (container terminals). One can thus compute the complete search trees from those 5 points once, and reuse them for subsequent requests. Constructing such a search tree takes $\Theta((|A| + |V|) \log |V|)$ time, and contains the shortest distance from all points to the root (one of these 5 points) and for each node a pointer to the next node on the shortest path to the root. Once these search trees are constructed, one can efficiently query a shortest path between any point and one of the roots, this will take a time proportional to the shortest path length (number of arcs on that path), which is in general much smaller than the number of nodes or arcs in the traffic network. Note that more advanced methods, such as the before mentioned contraction hierarchies, exist; however for the scale of our problem, the proposed method of a simple Dijkstra algorithm in combination with reusing shortest path trees is more than sufficient.

It is clear that by using a plain Dijkstra shortest path algorithm, other routing requests and real-time traffic information are not taken into account. To take into account

the current load on the traffic network when calculating optimal routes and avoid traffic jams, a delay metric or an additive resistance must be added. In the simulation, this will be done in the form of modifying the weights w_i of the arcs from simply the time it would take to traverse it, $\frac{L_i}{v_i}$, to a more general form:

$$w_i = \begin{cases} \frac{L_i}{v_i} & \text{if } n_i < n_i^{jam} \\ \frac{L_i}{v_i} \frac{n_i(c_i - n_i^{jam})}{n_i^{jam}(c_i - n_i)} & \text{if } n_i \geq n_i^{jam} \end{cases} \tag{13}$$

where n_i is taken to be $\min(n_i, c_i - 1)$ to avoid infinite numbers. The first case corresponds to the situation of free flow, whereas the second case is the travel time in the jammed phase where the weight is derived from the average speed as given by the fundamental diagram.

Note that the weight defined above depends on the state of the road segment and will change and evolve during the simulation or in real traffic situations, in other words, it is a function of time $w_i(t)$. Using equation (13) results in discrete jumps of the value of the weight. Also when computing routes one looks a the current weight of that arc and no information about the past evolution of the weights is used. In order to resolve this and smooth out the changes in weight, the exponential moving average (EMA) $\widetilde{w}_i(t)$ is kept for each arc; it is defined as

$$\widetilde{w}_i(t) = \frac{1}{\tau} \int_{-\infty}^{t} w_i(t') e^{-(t-t')/\tau} dt' \tag{14}$$

where τ is a parameters which controls the amount of smoothing and how far one looks to past weights; here it is set to $\tau = 10$ min. The EMA is updated every time the occupation n_i of an arc changes, i.e., when a vehicle enters or leaves the segment. Since w_i remains constant between these times, the EMA $\widetilde{w}_i(t)$ can be updated recursively in the following way:

$$\widetilde{w}_i(t) = (1 - \alpha) w_i(t) + \alpha \widetilde{w}_i(t_{\text{last}}) \quad \text{with} \quad \alpha = e^{-(t - t_{\text{last}})/\tau} \tag{15}$$

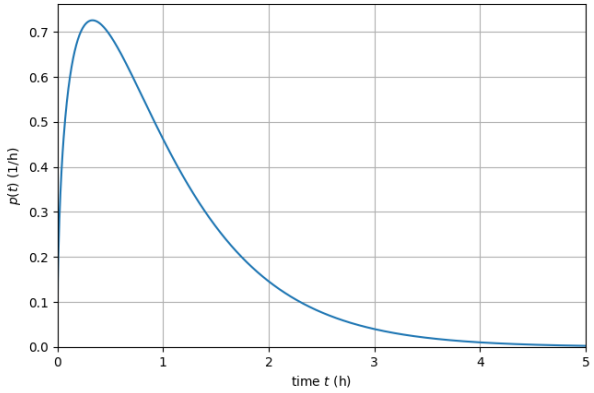
This means that only t_{last} and $\widetilde{w}_i(t_{\text{last}})$ have to be kept in memory for each arc and not all past values of the weights. Of course, recomputing the Dijkstra trees every time an arc weight is updated would be too costly and would defeat the whole purpose of precomputing them in the first place. That is why the effectively used weights and shortest paths are updated periodically after some predetermined time has passed. Here the routing weights and search trees are updated every 5 simulated minutes.

Initialisation of the simulation

The simulation aims to simulate the evolution of container transport by trucks on an average day in the Port of Antwerp. In order to initialise the simulation, a few things are taken care of. In the first place, the trucks are given random initial locations on the traffic network, representing the fact that trucks can be parked at the many roadside parking areas around the region. Following this, orders are generated for that day. As discussed earlier they have an origin and a destination that is either one of the five container terminals (with a probability proportional to their capacity as given in the main

article) and an industrial arc. These orders are randomly assigned to the order books of trucking companies, proportional to the number of trucks that each corresponding trucking company has in its fleet (on average 3 orders per truck). Finally, before the actual simulation starts, each truck is given a starting time, being the time when the drivers starts his/her actual day of transporting containers. The distribution of these starting times is somewhat smoothed, the reason for this is that not all drivers/trucks start at the same exact moment, moreover, this would result in much congestion in the traffic network. We assume that the starting times are distributed by a chi-square distribution χ_k with $k = 3$ and mean value 3,600 s, see Figure 12. Note that the reference point denoting the starting time of the simulation is somewhat arbitrarily set to zero. This does not represent the real time; it is assumed that the simulation starts when all terminals are opened.

Figure 12 Probability distribution of starting times of trucks/drivers, $\chi_{k=3}$ (see online version for colours)



After the initialisation of the model, the actual simulation can start and time begins to flow. Orders are allocated to trucks by a certain planning strategy and trucks carry out the orders by visiting the necessary location and terminal. The simulation ends once all orders are processed.