Transportation Research Part C 127 (2021) 103082

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Modeling and optimization of dedicated bus lanes space allocation in large networks with dynamic congestion

Dimitrios Tsitsokas [a], Anastasios Kouvelas [b], Nikolas Geroliminis [a,*]

[a] *Urban Transport Systems Laboratory, School of Architecture, Civil and Environmental Engineering, EPFL, CH-1015 Lausanne, Switzerland*
[b] *Institute for Transport Planning and Systems, Department of Civil, Environmental and Geomatic Engineering, ETH Zurich, CH-8093 Zurich, Switzerland*

ARTICLE INFO

Keywords:
Dedicated Bus Lanes (DBL)
Public transit priority
Dynamic traffic modeling
Large Neighborhood Search (LNS)
Optimization

ABSTRACT

Dedicated bus lanes provide a low cost and easily implementable strategy to improve transit service by minimizing congestion-related delays. Identifying the best spatial distribution of bus-only lanes in order to maximize traffic performance of an urban network while balancing the trade-off between bus priority and regular traffic disturbance is a challenging task. This paper studies the problem of optimal dedicated bus lane allocation and proposes a modeling framework based on a link-level dynamic traffic modeling paradigm, which is compatible with the dynamic characteristics of congestion propagation that can be correlated with bus lane relative positions. The problem is formulated as a non-linear combinatorial optimization problem with binary variables. An algorithmic scheme based on a problem-specific heuristic and Large Neighborhood Search metaheuristic, potentially combined with a network decomposition technique and a performance-based learning process for increased efficiency, is proposed for deriving good quality solutions for large-scale network instances. Numerical application results for a real city center demonstrate the efficiency of the proposed framework in finding effective bus lane network configurations; when compared to the initial network state they exhibit the potential of bus lanes to improve travel time for car and bus users.

1. Introduction

Mitigating traffic congestion caused by excessive demand over limited capacity is a primary goal for transport planners in rapidly developing cities. With urban road space expansion being often impossible, extremely costly or even inefficient, due to the effect of induced demand (Toth, 2007), optimizing public transport services seems a promising and sustainable strategy to alleviate traffic congestion (Diakaki et al., 2014). Public Transit Priority (PTP) strategies serve this exact purpose, with a long-term goal of motivating commuters to opt for public transport instead of private car in the city center, leading to reduced car demand and more efficient road space use.

Among various PTP measures, Dedicated Bus Lane (DBL) installation has seen wide implementation due to its effectiveness in reducing bus delays caused by traffic interactions, while being inexpensive and relatively fast to implement. Space separation also results in increased punctuality and reliability for the bus service. However, space reserved for transit vehicles is taken from non-transit traffic, which can potentially induce congestion due to local capacity reduction, e.g. in neighborhoods of infrequent bus service or

* Corresponding author.
E-mail addresses: dimitrios.tsitsokas@epfl.ch (D. Tsitsokas), kouvelas@ethz.ch (A. Kouvelas), nikolas.geroliminis@epfl.ch (N. Geroliminis).

https://doi.org/10.1016/j.trc.2021.103082
Received 30 September 2020; Received in revised form 2 February 2021; Accepted 3 March 2021
Available online 1 May 2021

inflexible non-transit demand (Dahlgren, 1998), indicating that location selection plays an important role in the efficiency of DBL strategy.

Balancing the trade-off between public transit enhancement and general traffic disturbance in an optimal way when designing a DBL network is a considerably challenging task. This is due not only to the complexity included in multi-modal network traffic modeling, but also the competitive nature of the problem, which requires accurate prediction of travelers' reactions concerning mode and route preferences; these can be significantly influenced by any infrastructure change (such as a DBL network installation) and disturb the established traffic equilibrium. Due to its high complexity, most researchers have addressed the DBL optimal design problem by assuming static traffic conditions and focusing on the bi-level structure of the problem. However, congestion propagation due to spillback effects of queues is not considered in the DBL network design process when static traffic modeling and assignment is assumed, in spite of being closely interrelated to DBL relative locations. Moreover, most applications are done in small-scale networks with only a few roads and intersections.

In this work we address the problem of optimal DBL location selection for a given network with known bus routes and frequencies, by integrating a dynamic, link-level macroscopic traffic model with queuing characteristics into an optimization framework that complies with the bi-level nature of the problem. We propose an effective algorithmic scheme based on Large Neighborhood Search (LNS) metaheuristic and problem-specific heuristics for finding near-optimal solutions for considerably large networks. The main contributions of the paper concentrate on the integration of dynamic traffic modeling in this complex optimization problem, which can properly capture the dynamic characteristics of congestion propagation due to queue spillbacks, as well as on the applicability of the proposed optimization algorithms in large-scale network instances. The effective combination of both of these elements for the solution of the DBL allocation problem, to the best of our knowledge, has not been tackled before. Additionally, effective application of LNS metaheuristic for the optimal DBL allocation problem is demonstrated in this work, making the proposed method a powerful alternative to Genetic Algorithms (GA) that are typically used in the literature (e.g. Yao et al., 2012; Sun and Wu, 2017; Zhao et al., 2019). A network decomposition technique is also proposed as an additional component to LNS repair process, with the aim of accelerating the solution process in very large network instances. The proposed framework is applied to the road network of San Francisco business district and the generated results and algorithmic performance are reported and discussed.

The remainder of the paper is organized as follows: Section 2 briefly reviews the existing literature and justifies the motivation for the present work. Section 3 formally introduces the problem and describes the proposed modeling and optimization framework, which integrates a dynamic link-based macroscopic traffic model with an iterative mode choice adjustment process. The mathematical formulation of the optimization problem is also presented here. The proposed optimization algorithms, based on a problem-specific heuristic and LNS, are presented in Section 4. Algorithm performance and generated solutions for a real case study of San Francisco network are discussed in Section 5, where the best derived DBL plans are compared to the initial network state (no DBL) in terms of simulated total travel time. Finally, Section 6 concludes the paper providing some useful insights.

2. Background

Various PTP measures have been proposed in the literature, ranging from infrastructure improvements, such as Bus Rapid Transit (BRT) systems (Levinson et al., 2003; Deng and Nelson, 2011; Wirasinghe et al., 2013), application of smart technologies, such as Transit Signal Priority (TSP), pre-signals, queue-jumper lanes or perimeter control (Farid et al., 2015; Guler et al., 2016; Guler and Cassidy, 2012; Christofa et al., 2013; Christofa et al., 2016; Ampountolas et al., 2017), or planning for exclusive road space (e.g. bus-only lanes, bus-lanes with intermittent priority or dynamic priority lanes, see Viegas and Lu, 2001; Eichler and Daganzo, 2006; Anderson and Geroliminis, 2020). The great potential of DBL, especially, is demonstrated in numerous studies performed in the last few decades, addressing the topic from several perspectives and in different scales.

Microscopic traffic simulation is often utilized for the evaluation of roads and road networks with existing DBL systems (e.g. Choi and Choi, 1995; Wei and Chong, 2002; Waterson et al., 2003), as well as for alternative scenario analysis for future development (e.g. Shalaby, 1999; Abdelghany et al., 2007; Chen et al., 2010; Arasan and Vedagiri, 2010). Abdelghany et al. (2007) developed a dynamic assignment and simulation framework for the evaluation and planning of BRT systems by integrating exclusive road space for buses. Stirzaker and Dia (2007) performed microsimulation analysis for a real corridor in Brisbane, Australia, in order to assess the impact of setting a DBL or High Occupancy Vehicle (HOV) lane, aiming at road use maximization. Khoo et al. (2014) also utilized microscopic simulation for the DBL allocation and scheduling problem in a bi-objective formulation and applied GA to solve it. Farid et al. (2018) developed analytical models based on the kinematic wave theory for person-based evaluation of the combined effect of DBL, Queue-Jumper Lanes and TSP strategies in signalized intersections using microsimulation. Numerical models have been applied by Basso et al. (2011) for mode choice and car-bus interactions modeling in a synthetic road, in order to compare different congestion management policies, including congestion pricing and transit subsidization, while optimizing bus operations and DBL provision for maximizing social welfare. Although microscopic simulation may provide an analysis of increased accuracy compared to macroscopic or empirical models, it is often too expensive computationally to be included in an optimization framework, especially for large networks. Most of these studies involve local scale DBL applications in specific arterial roads, freeways, or small corridors, and their outcomes result from scenario evaluation, rather than a well-established optimal design process.

Going a step further, strategic planning methodologies for DBL allocation have also been proposed, with a special focus on handling passenger response concerning mode and route choices. Bi-level programming is typically utilized, where a social optimum objective for the DBL location assignment is decided by traffic planners in the upper layer, and the respective user reactions to these decisions are determined in the lower level, given that users seek to maximize their personal utilities by making appropriate mode, route, and departure time selection. Mesbah et al. (2011) have modeled the DBL location selection problem as a Stackelberg competition and applied decomposition and cutting planes techniques for solving the corresponding bi-level program. A similar formulation was later proposed by Yu et al. (2015), where column generation, branch-and-bound, and successive averages have been applied for deriving the solution. Miandoabchi et al. (2012) have formulated a discrete bi-modal Network Design Problem (NDP) as a bi-level program with equilibrium constraints and several decision variables, including new road construction, lane addition, lane direction assignment, and bus-only lane assignment, which they have addressed with hybrid metaheuristic algorithms. Yao et al. (2012) have formulated a bi-level programming model for DBL setting and bus frequency optimization, considering an integrated network equilibrium model in the lower level, and used GAs for optimization. Sun and Wu (2017) and Zhao et al. (2019) also proposed bi-level programming structures while focusing on multiple objectives and operational intersection dynamics and applied GAs for the solution process. While bi-level programming integrates modeling of passengers reactions in the strategic planning phase, it often leads to highly complex problem formulations requiring excessive computational resources, which, again, makes the application practically impossible to large-scale networks. Moreover, steady-state traffic conditions are typically assumed in such formulations, with link travel times being calculated by empirical relationships on the basis of "Bureau of Public Roads" (BPR) functions proposed in the Highway Capacity Manual (NRC, 2010). However, static traffic assignment cannot capture the effects of queue formation and spillbacks related to backward propagation of congestion, which can be highly correlated to DBL presence, resulting to potentially unrealistic estimation of network performance.

Interesting studies on urban road space management based on macroscopic traffic modeling also exist, founded on the concept of the Macroscopic Fundamental Diagram (MFD), also known as Network Fundamental Diagram (NFD). Gonzales et al. (2010) utilize the MFD concept for urban space management and distribution between competing modes, and show that reserved lanes for transit or high occupancy vehicles can reduce traffic related delays even for non-prioritized vehicles. Zheng and Geroliminis (2013) propose the concept of a multi-modal MFD to capture the dynamic interactions between competing modes in multi-region cities, and develop a framework for road space allocation that optimizes passenger throughput. Geroliminis et al. (2014) and Chiabaut (2015) study the concept of passenger MFD (p-MFD) as a useful tool to integrate in the development of optimal transit operation strategies. Zhang et al. (2018) propose an MFD-based framework for optimizing road capacity management together with transit operations, by integrating mode choice and dynamic user equilibrium in modeling dynamics. More recently, Anderson and Geroliminis (2020) study the impact of a controlled bus lane, which allows regular vehicles to enter under certain conditions, according to a dynamic control framework based on MFD modeling. Nevertheless, although MFD-based approaches are able to capture dynamic interactions between competing modes, they do this in an aggregated way for a region or arterial road and cannot drive decision making processes on the link-level of a city network.

To respond to the above limitations of existing research, the current work addresses the problem of optimal DBL location selection by proposing a modeling framework on the basis of a dynamic urban traffic model with queuing characteristics, typically used in model-based control applications. This type of traffic representation can capture the topological variations of congestion propagation in the road level and evaluate the impact that candidate DBL location schemes will have in the resulting congestion patterns. At the same time, it is easily integrated in an optimization framework. To the best of our knowledge, there have been particularly few attempts in the same direction. Li and Ju (2009) have proposed a modeling framework based on a point-Q model, where a Variational Inequality formulation is integrated to capture the mode, route, and departure time choices of passengers as a result of DBL presence, but only tested two alternatives in small network instances. More recently, Bayrak and Guler (2018) have also used a dynamic traffic model with horizontal queues to identify the best DBL plan and used GA to solve the optimization problem for a small network instance.

With the aim of proposing a method simple enough to be applied to large networks, we simplify the typical bi-level programming structure by removing the necessity of solving interconnected optimization problems in the lower lever to account for passenger mode and route responses. Instead, we utilize a simple Logit model to capture commuter mode preferences based only on the estimated travel time per mode, while we assume that route choice patterns remain unaffected by DBL introduction. Even though dynamic traffic assignment model could be integrated for a more realistic representation of route choices, it might add significant computational complexity in the problem without guarantee of significant increase in accuracy. Nevertheless, the developed optimization framework is quite general and could be utilized with enhanced modeling efforts in the future. While GA is a typical method choice for this type of problems, in our case, dynamic traffic modeling significantly increases complexity, thus population based metaheuristics become less efficient. Instead, Local search (LS), often introduced in a Simulated Annealing (SA) framework, has shown promising results in tackling combinatorial optimization problems referring to location or resource allocation (e.g. Zockaie et al., 2018). More specifically, LNS, firstly proposed by Shaw (1998), has been successfully applied in Vehicle Routing Problems (VRP), but also in public transport scheduling, facility location and logistics, due to its ability to efficiently explore large solution spaces by using heuristics. A comprehensive review on the various applications of LNS and its more recent variant A-LNS can be found in Pisinger and Ropke (2019). In this work we propose a set of algorithms based on a problem-specific heuristic and LNS metaheuristic, that can be combined with a simulation-based, learning algorithm. The latter tries to estimate the probability of every link in improving system performance when one lane is converted to DBL. These probabilities, in the form of link scores, can then drive the search process of LNS algorithm and increase its performance by implicitly considering all physical network characteristics in the solution construction process.

3. Problem description and modeling framework

3.1. Problem description

Consider an urban traffic network facing high levels of congestion during peak-hour. The geometrical, topological, and traffic control characteristics of the network are considered known. Two modes of transport are available in this network: buses and private cars. The operational characteristics of the bus system (routes, frequencies, and bus stop positions), as well as the average passenger occupancy for all bus lines in all roads over several time-periods of the day are also known. A deterministic time-dependent Origin–Destination demand matrix feeds the network with private car flow. The routing choices of vehicles are assumed known, in the form of time-dependent turning ratios, for all intersections approaches. Assuming that, for the purpose of improving mobility in the network by prioritizing public transport, a fraction of general purpose road space is to be given for DBL installation, we seek to decide upon the best possible spatial DBL distribution in the network, with the aim of achieving optimal system performance from a passenger perspective. The optimal subset of network roads (links) needs to be identified, where the right-most lane would be transformed to DBL, in order to achieve minimum total passenger travel time. The DBL temporal assignment is considered constant during the study period, representative of the morning or the evening peak, and no attribute of the bus system operational characteristics (e.g. headway) is modified. The mathematical modeling constructed to address this problem is described in the following sections.

3.2. Network traffic model

Traffic evolution in the network is simulated by utilizing a macroscopic, link-based traffic model inspired by queuing theory, and built on the basis of two existing models, the “S-Model” (Lin et al., 2011; Lin et al., 2012) and the “Store-and-Forward” (SaF) model (Aboudolas et al., 2009; Kouvelas et al., 2014), which are often used in model-based control applications and meet our requirements for efficiency and simplicity. While earlier versions of SaF may consider only vertical queues, we utilize the adjusted version of SaF (Aboudolas et al., 2009) that considers horizontal (capacitated) queues for proper handling of spillbacks, and expand it by integrating the “moving” and “queuing” vehicle separation of “S-Model”, which provides increased accuracy in travel time calculation by properly integrating link lengths. Consisting of a mathematical formulation based on a time-discretized flow conservation equation, the model dynamically updates the number of vehicles per link, according to road space availability, traffic signals, saturated and unsaturated flows. Its mathematical formulation is described below.

A traffic network is represented as a directed graph $G = (N,Z)$, consisting of a set of nodes (junctions) N and directed links (roads) Z. Every link $z \in Z$ is a unique, one-way connection between a pair of nodes (s_z, e_z), where $s_z, e_z \in N$ denote the upstream (start) and downstream (end) nodes of link z, respectively. Moreover, every link z is associated to a set of upstream and downstream links, U_z and D_z, respectively. Note that, every link $i \in U_z$, i.e. belonging to the set of upstream links of z, is directly connected to z and vehicles are allowed to move from i to z; the same applies for the relation between link z and any of its downstream links $j \in D_z$. The system state at every time step k is described by the number of vehicles per link $z \in Z$, denoted as $x_z(k)$. The dynamic equations are described below.

$$x_z(k) = m_z(k) + w_z(k) \tag{1}$$

$$m_z(k+1) = m_z(k) + T\left(u_{\mathrm{VQ}_z}(k) + \left(1 - t_{z_0}(k)\right)\sum_{\forall i \in U_z} u_{iz}(k) - a_z(k)\right) \tag{2}$$

$$w_z(k+1) = w_z(k) + T\left(a_z(k) - \sum_{\forall i \in D_z} u_{zi}(k)\right) \tag{3}$$

$$\forall z \in Z,\ \ k = 1, 2, \dots, K-1.$$

In the above equations, $z \in Z$ is the link index, k is the time-step index, T denotes the discrete time-step duration, and KT is the total simulation period. Eq. (1) states that the number of vehicles inside link z at time step k, denoted as $x_z(k)$, is composed by the sum of “moving” vehicles $m_z(k)$, i.e. vehicles moving with free-flow speed in the non-occupied part of the road, and “queuing” vehicles $w_z(k)$, i.e. vehicles already queuing upstream the intersection at the end of the link. The dynamics of moving and queuing vehicles for every link z are described by Eqs. (2) and (3), respectively, where u_{ij} denotes the transfer flow from upstream link i to downstream link j, where $i,j \in Z$ and $i \in U_j \equiv j \in D_i$; a_z refers to the flow arriving at the tail of the queue inside link z, i.e. flow leaving the “moving” and joining the “queuing” part of z; u_{VQ_z} denotes the outflow of the virtual queue (explained below) of link z, which relates to the newly generated demand at link z; finally, t_{z_0} denotes the time-dependent fraction of the incoming vehicle flow of z that end their trip in z. These vehicles are assumed to exit the network just upon entering their destination link.

Transfer flow $u_{zi}(k)$ between any pair z, i of consecutive links, with $z, i \in Z, z \in U_i \equiv i \in D_z$, is calculated by Eq. (4), by taking into account the current traffic signal state of the approach, indicated by binary variable $\eta_{zi}(k)$, storage capacity c_i and current state x_i of the receiving link i, and the saturated or unsaturated flow of the approach z–i (depending on current state of queue in sending link z). For signalized intersections, $\eta_{zi}(k)$ is equal to 1 if approach z–i takes green light at time-step k, and zero otherwise (Eq. (5)). For non-signalized intersections, $\eta_{zi}(k)$ can be constant $\forall k$, according to the priority of movements and geometry of merging links z and i. Storage capacity c_z, referring to the maximum number of vehicles that can be in z simultaneously, is given by Eq. (6), where l_z denotes the number of lanes in link z, L_z the link length, and l_{veh} the average vehicle length. Saturation flow of any link z is assumed to be equal to 1800 veh/h/lane (Eq. (8)). However, this number can be adjusted to the specific network or link, based on real flow data. Saturated flow of an approach z–i is the minimum between the saturated flows of consecutive links z, i, by considering the number of lanes that are available for this approach in each link. This is shown in Eq. (7), where $l_{zi} \leqslant l_z$ denotes the number of lanes in sending link z that allow moving to downstream link i, according to existing traffic regulations. We assume that in the receiving link, all lanes are available to all arriving vehicles, independently of their link of origin. Unsaturated flow, calculated by the right term of the minimum function in Eq. (4), refers to the flow allowing all vehicles currently queuing to move from z to i in one time-step. The time-dependent turn ratio that corresponds to approach z–i, denoted as $t_{zi}(k)$, refers to the fraction of the current queue in link z that will move to link i. Hence, according to Eq. (4), outflow of any approach z–i at any time-step k is zero when the traffic light is red or when unoccupied space in receiving link is less than the maximum flow that the link can receive in one time-step (queue almost occupies the whole link); otherwise, it is equal to the approach's saturated or unsaturated flow, depending on the queue length in upstream link z.

$$u_{zi}(k) = \eta_{zi}(k) \times \begin{cases} 0 & \text{if } c_i - x_i(k) \leqslant S_i T \\ \min\left(S_{zi}, \dfrac{(w_z(k) + a_z(k))\ t_{zi}(k)}{T}\right) & \text{else} \end{cases} \tag{4}$$

$$\eta_{zi}(k) = \begin{cases} 1 & \text{if } z \rightarrow i \text{ has right-of-way at time-step } k \\ 0 & \text{else} \end{cases} \tag{5}$$

$$c_z = \frac{l_z\ L_z}{l_{\mathrm{veh}}} \tag{6}$$

$$S_{zi} = 1800 \cdot \min(l_{zi}, l_i)\ \ (\mathrm{veh/h}) \tag{7}$$

$$S_z = 1800 \cdot l_z\ \ (\mathrm{veh/h}) \tag{8}$$

$(4), (5),$ and $(7)\ \forall z \in U_i, \forall z, i \in Z;\ (6)$ and $(8)\ \forall z \in Z.$

In case of high congestion, links may reach their storage capacity and newly generated demand may not be able to be received due to space limitation. As the model forces incoming flows from upstream links to wait in queues in such cases, the same should happen to the newly generated inflow. Therefore, similarly to most traffic simulation models, a "virtual queue" is assumed upstream of every link that serves as entry to the network (can be origin of trips). At first, newly generated flow joins the virtual queue and remains there as long as the entrance link is full. This effect is included in the model by means of a virtual link with infinite storage capacity, assumed upstream of every origin link. As the time spent in virtual queues is important for system performance and should be considered when calculating the overall travel time of vehicles, dynamics of virtual queues, similarly to actual links, are updated based on the following equations:

$$x_{\mathrm{VQ}z}(k+1) = x_{\mathrm{VQ}z}(k) + T(d_z(k) - u_{\mathrm{VQ}z}(k)) \tag{9}$$

$$u_{\mathrm{VQ}z}(k) = \begin{cases} 0 & \text{if } c_z - x_z(k) \leqslant S_z T \\ \min\left(S_z, \dfrac{x_{\mathrm{VQ}z}(k)}{T}\right) & \text{else} \end{cases} \tag{10}$$

$\forall z \in Z.$

In Eq. (9), $x_{\mathrm{VQ}z}$ denotes the number of vehicles in the virtual queue of link $z \in Z$, d_z the newly generated demand of trips starting in z, and $u_{\mathrm{VQ}z}$ the virtual queue's outflow towards link z. The latter is calculated similarly to any approach outflow u_{zi}, by Eq. (10).

Flow arriving at the end of a queue at time-step k, denoted as $a_z(k)$, is calculated based on the tail's current position, which depends on the number of queuing vehicles $w_z(k)$. At every time-step k, the number of discrete time-steps required for a vehicle to travel the distance between link start node and current queue end (with free-flow speed v_{ff}), is calculated by the term inside $\mathrm{ceil}(\cdot)$ in Eq. (11). By subtracting this time from the current time-step k, we can determine the discrete time-step by which, the total link inflow since simulation started must have already reached the current queue end. This time point is denoted as $\rho_z(k)$ and is non-decreasing with k, as shown by Eq. (11). The current arriving flow $a_z(k)$ is then computed as the difference between the cumulative link inflow that is calculated up to time-step $\rho_z(k)$ minus the cumulative link inflow up to time-step $\rho_z(k-1)$, which is considered to have already arrived at the previous time-step. This process is described by Eq. (12).

$$\rho_z(k) = \max\left(\rho_z(k-1), k - \mathrm{ceil}\left(\frac{(c_z - w_z(k)) l_{\mathrm{veh}}}{l_z v_{\mathrm{ff}} T}\right)\right), k \geqslant 1, \rho_z(0) = 0 \tag{11}$$

$$a_z(k) = \sum_{j=1}^{\rho_z(k)} \sum_{\forall i \in U_z} u_{iz}(j) - \sum_{j=1}^{\rho_z(k-1)} \sum_{\forall i \in U_z} u_{iz}(j) \tag{12}$$

$\forall z \in Z.$

It should be noted that the paths of vehicles circulating in the network are not known to the model; however, the impact of route choice in propagation of congestion is expressed through turning ratios $t_{ij}(k)$ and exit ratios $t_{z_0}(k)$. These can either be constant for the entire simulation time or vary between time-steps in the same way as traffic patterns vary throughout the day.

At every time step $k = 1, 2, \ldots, K$, the states (number of moving, queuing, and total vehicles) of all links $z \in Z$ and virtual queues of origin links are updated according to Eqs. (1)–(12). Necessary inputs include the detailed network representation (connectivity, length and number of lanes per link, right-of-ways), initial state of the system, i.e, $x_z(0), x_{\mathrm{VQ}z}(0), \forall z \in Z$, dynamic demand $d_z(k)$ at all origin links, traffic signal plans, time-dependent turn ratios $t_{ij}(k)$, and link exit rates $t_{z_0}(k)$. The above formulation, by considering link capacities, available road space and traffic signals for outflow calculation, can properly capture the spatio-temporal propagation of queues and spillbacks.

3.3. Decision variables

The effect of transforming existing general-purpose lanes to bus-only lanes, in terms of traffic flow modeling, is equivalent to reducing the number of available lanes for car traffic in the respective roads. Assuming that, in every link, one lane, at most, can be transformed to DBL, we define a binary variable y_z, for every link $z \in Z$, which is equal to 1 if a DBL is installed in link z, and 0 otherwise (Eq. (13)). The number of general purpose lanes l_z, used for determining storage capacity and saturation flow, is then replaced in all equations of the traffic model by the difference $l_z - y_z$.

$$y_z = \begin{cases} 1, & \text{if a DBL is assigned to link } z \\ 0, & \text{otherwise} \end{cases}, \forall z \in Z. \tag{13}$$

Variables $y_z,\ z \in Z$, constitute the decision variables of the optimization problem. A feasible solution to the problem is represented as a binary vector $\mathbf{y}$ of dimension equal to the number of links $z \in Z_f \subseteq Z$, where Z_f denotes the set of candidate links for DBL installation, i.e., all links that satisfy a number of case-specific predefined criteria, such as having at least two lanes and belonging to the path of at least one bus line. Obviously, $y_z = 0, \forall z \notin Z_f$.

3.4. Total travel time estimation

The objective of the problem is to maximize system performance through appropriate DBL allocation. This is translated into minimizing total travel time of all passengers, or Passenger Hours Travelled (PHT), travelling by either car or bus, during the simulated time and for a specific demand scenario, which is given by the following set of equations:

$$\mathrm{PHT}_c = \sum_{z \in Z} \sum_{k} \left(x_z(k) + x_{\mathrm{VQ}z}(k)\right) \xi T \tag{14}$$

$$\mathrm{PHT}_b = \sum_{z \in Z} \sum_{k} \sum_{l \in B} \left(\left((1 - y_z) \frac{L_z}{v_z^{\mathrm{c}}(k)} + y_z \frac{L_z}{v_{\mathrm{ff}}} \right) P_z^l(k) + D_z^l(k) \right) T \tag{15}$$

$$v_z^{\mathrm{c}}(k) = \min \left(v_{\mathrm{ff}}, \frac{\sum\limits_{j=k-t_w+1}^{k} u_z(j) L_z}{\sum\limits_{j=k-t_w+1}^{k} x_z(j) t_w} \right), k > t_w \tag{16}$$

$$D_z^l(k) = \delta_z^l \left(\beta_z^l P_z^l(k) s_p + s_f f_l(k) \right) \tag{17}$$

$$\mathrm{PHT} = \mathrm{PHT}_c + \mathrm{PHT}_b \tag{18}$$

Total PHT for car users, denoted as PHT_c, is derived from the total time that cars spent inside the network and in virtual queues. It is calculated by summing the queues in all links over time, as shown by Eq. (14), where ξ denotes the average car occupancy. Regarding bus travel time, since the traffic model only monitors car flows, it has to be estimated indirectly. This is done by assuming that buses travel in free-flow conditions inside links with DBL ($y_z = 1$), while in links without DBL ($y_z = 0$) we assume that buses travel either with speed of cars, which can be estimated using link queues and outflows over a specific time window, or with free-flow speed v_{ff} if this is smaller, as shown in Eq. 16. Note that all bus stops are assumed to provide bus bays outside of traffic lanes with enough capacity to minimize traffic disturbance during boarding and alighting of passengers. Total PHT for bus passengers, denoted as PHT_b, is calculated by Eq. (15), where notation is as follows: B is the set of bus lines running in the network; l is the bus line index; $v_z^{\mathrm{c}}(k)$ denotes the estimated car speed in link z at time step k, which is calculated based on link queue and outflow, according to Eq. (16); $P_z^l(k)$ is the average flow of bus passengers (pax/h) traveling on-board line l buses through link z at time step k; and $D_z^l(k)$ is the dwell time of buses for line l at the bus stop in link z at time step k. Therefore, according to y_z value, travel time of buses inside link z is estimated by assuming either free-flow speed v_{ff} in case of DBL presence ($y_z = 1$) or an estimation of car speed v_z^{c} in case of no DBL presence ($y_z = 0$). Consequently, if a link gets congested and its car outflow drops, estimated bus speed in this link will be affected accordingly. The values $P_z^l(k)$ can be estimated as $P_z^l(k) = f_l(k) Pb_z^l(k)$, where $f_l(k)$ is the frequency of line l at time step k, and $Pb_z^l(k)$ the average number of passengers per bus of line l inside link z at time-step k. These values can be derived from measurements performed by the bus operator.

Car speed in link z, v_z^{c}, is estimated by Eq. (16), as the minimum between the assumed free-flow speed v_{ff} and the fraction of total distance travelled by cars inside link z during a time window of t_w time-steps (prior to time-step k), divided by the total time spent by all cars inside link z in the same time window. It should be noted that in the case where link z remains empty during the entire time window, speed is equal to free-flow speed. Service time at bus stops is calculated according to Eq. (17), where δ_z^l is a binary indicator about bus stop existence in link z for buses of line l; β_z^l is the average fraction of bus occupancy of line l inside link z that corresponds to the sum of boarding and alighting passengers for the bus stop in link z; s_p is the time delay per boarding/alighting passenger; s_f is a fixed delay per bus stop. This empirical relation, formulated based on a survey reported by Meng and Qu (2013), considers as $s_p = 1.5$ sec per boarding/alighting passenger plus a fixed delay of $s_f = 4$ sec per stop. The average bus passenger occupancy of bus lines over time and space $Pb_z^l(k)$, as well as bus frequencies $f_l(k)$, and approximate fractions β_z^l are considered as known inputs in the context of this work. Total travel time of all passengers, denoted as PHT is the sum of PHT_c and PHT_b (Eq. (18)).

3.5. Mode and route choices

DBL introduction is expected to change experienced travel times and congestion patterns for both cars and buses in several parts of the network. Consequently, user choices in terms of mode and routes may be affected by DBL location assignment and car/bus shares and turn ratios may change accordingly. Therefore, the evaluation process of every candidate solution should account for these changes. This is why, in most related works, the problem is formulated as a bi-level program, similar to a Stackelberg competition, where traffic managers act as leaders, deciding on the best DBL plan to optimize system performance, and users act as followers, deciding upon their mode and route preferences, given the decisions of the traffic managers, with the aim of optimizing their personal overall travel utility (often including in-vehicle travel time, out-of-pocket cost, access time and other attributes).

While a bi-level programming approach might be possible when static traffic assignment is used, in our case, since our dynamic macroscopic model uses turning ratios for traffic distribution in intersections, exact trip paths are not known. Updating turn ratios for every new DBL configuration through a suitable traffic assignment process might improve the accuracy of system performance estimation, but, at the same time would add an unrealistic computational burden in the optimization process. This is why a typical bi-level programming structure, where the optimal solutions of lower level optimization problems are included in the constraints of the optimization problem of the upper level, would drastically increase problem's complexity, making the method inapplicable to large networks, which is one of our primary objectives. Seeking to avoid this effect, we model mode choice in a simpler, aggregated way, by using a Logit model, while we assume, for simplicity, that route choices of cars remain unaffected by DBL installation. While we understand that this assumption might affect the accuracy of the evaluation process for candidate solutions during optimization, we believe that its impact would not significantly alter the optimal solution, especially in large networks, where highly complex models are not applicable. Nevertheless, this assumption could be removed in the future by integrating, in the same modeling framework, a turn ratio update process, resulting from suitable traffic assignment considering current DBL locations, which would be efficient and simple enough, in order to not drastically increase the overall computational cost. However, this addition exceeds the scope of the present work and is currently omitted.

In order to address the expected variations in mode shares related to the specific DBL plan under consideration, we utilize a simple Logit model embedded in an iterative scheme of traffic simulations for evaluation of every candidate DBL plan. The process is depicted in Fig. 1. The selected DBL plan is provided as input to the urban traffic model, together with all other exogenous inputs, including

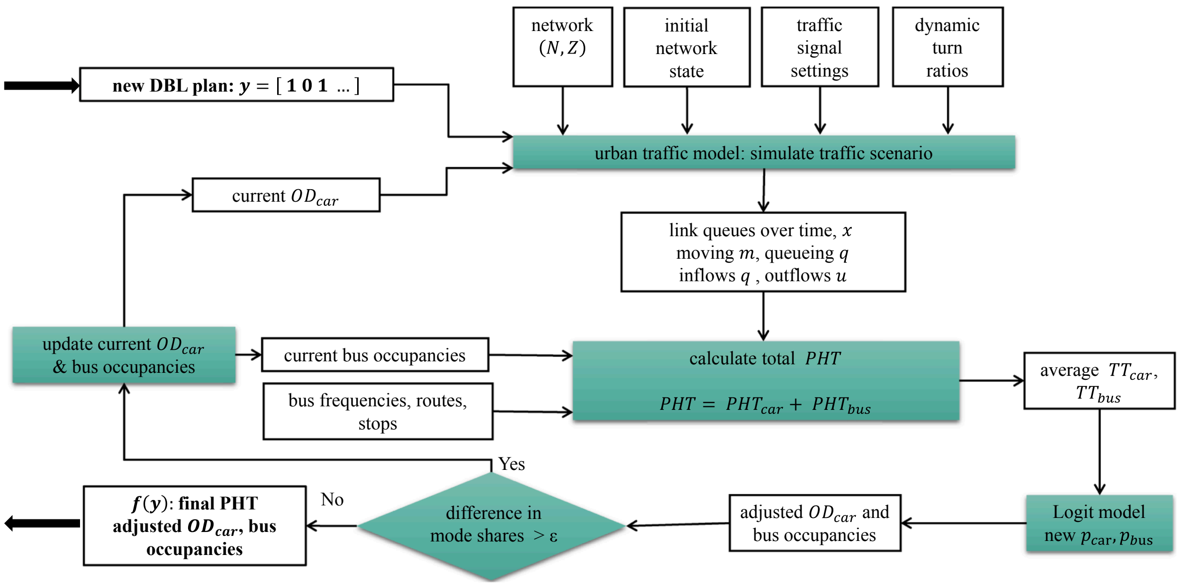

network (N, Z)
initial network state
traffic signal settings
dynamic turn ratios
new DBL plan: $y = [1\ 0\ 1\ ...]$
urban traffic model: simulate traffic scenario
current OD_{car}
link queues over time, x moving m, queueing q inflows q, outflows u
update current OD_{car} & bus occupancies
current bus occupancies
bus frequencies, routes, stops
calculate total PHT
$PHT = PHT_{car} + PHT_{bus}$
average TT_{car}, TT_{bus}
Yes
No
$f(y)$: final PHT adjusted OD_{car}, bus occupancies
difference in mode shares $> \varepsilon$
adjusted OD_{car} and bus occupancies
Logit model new p_{car}, p_{bus}

Fig. 1. Evaluation process of any newly formed candidate solution **y** (DBL plan) with mode choice adjustment, for calculation of the respective cost $f(\mathbf{y})$ (PHT).

current car demand and bus ridership for all lines. Based on queue evolution inside links and bus operational characteristics, we estimate the total PHT for car and bus passengers, from which, by dividing by the number of passengers and average trip length per mode, we obtain the average time per unit distance travelled for every mode (inverse of average speed). Then, the percentage of users opting for car or bus is calculated based on the utility of each mode, according to Logit model. The new mode shares are translated to new car demand and bus occupancies, which are compared to the previous ones. If their difference is above a specified threshold, a new traffic simulation is performed by considering the derived car and bus demand information. For simplicity, the utility function only considers travel time per kilometer of trip plus a constant term; however, more attributes can be included, such as out-of-pocket costs, accessibility, parking availability, and others. Utility per mode and percentage of users for each mode over all network users are calculated according to the following equations:

$$U_m = ASC_m + \beta_m \cdot UTT_m \tag{19}$$

$$p_m = \frac{e^{U_m}}{e^{U_c} + e^{U_b}} \tag{20}$$

In (19) and (20), we use m as the mode symbol, i.e. car (c) or bus (b), UTT_m denotes the inverse of average speed for mode m, i.e. travel time per unit distance, ASC_m and β_m are model parameters, and p_m denotes the percentage of total number of users opting for mode m, according to Logit model. Average car trip length and number of bus passengers can be obtained from a preliminary microsimulation analysis and are assumed constant. The adjustment of car demand and bus ridership per line in order to comply with a modified mode share, is performed in a uniform way across all OD pairs and bus lines, by maintaining the initial distribution of trips over OD pairs and bus lines. This is based on the assumption that population is homogeneous and all trips can be performed by both modes.

3.6. Optimization problem formulation

The objective of the problem is to identify the DBL layout that is expected to lead to minimum total passenger travel time. For every candidate road, a decision about whether a bus-only lane will be installed (by conversion of a general-purpose lane) or not, needs to be made. It should be noted that the total road space to be devoted to bus-only lanes may be predefined, especially in cases where road space is quite limited, or it may be indifferent/unlimited. To address different cases regarding total DBL length, different variants of the optimization problem can be constructed on the same framework: in one variant, the objective function may consist only of the PHT term (Eq. (18)) and a constraint may be considered, to ensure that feasible solutions have a total length close enough to (or simply less than) a predefined target length; in a second version, a penalty cost term, proportional to the total DBL solution length, can be included in the objective function to penalize excessive use of road space, and no additional constraint regarding total length is considered. This modification of the objective function aims at driving the search towards solutions of more efficient road space use than others, in the sense that they improve traffic performance while considering the amount of road space devoted to DBL. The value of the cost per unit length, denoted as γ, can be related to actual installation costs (e.g. horizontal signalization, mechanical infrastructure etc.), or it can simply reflect an approximate generalized cost associated to car traffic disturbance. Obviously, higher values of γ will lead to solutions with less DBL road space usage. The detailed mathematical formulation of the optimization problem is presented below:

$$\begin{aligned}\min_{y_z,\forall z\in Z_f} \quad & \sum_{z\in Z}\sum_{k}\sum_{l\in B}\left(\left((1-y_z)\frac{L_z}{v_z^{c}(k)}+y_z\frac{L_z}{v_{\text{ff}}}\right)P_z^l(k)+\delta_z^l\left(\beta_z^l P_z^l(k)s_p+s_f f_l(k)\right)\right)T+ \\ & \sum_{z\in Z}\sum_{k}(x_z(k)+x_{\text{VQ}z}(k))\xi T+\gamma\sum_{z\in Z}y_z L_z\end{aligned} \tag{21}$$

subject to:

Equations (1), (2), (3), (4), (5), (9), (10), (12)

$$S_{zi} = 1800\cdot\min(l_{zi}-y_z\ rl_{zi},\ l_i-y_i),\ \ (\text{veh/h}) \tag{22}$$

$$c_z = \frac{(l_z-y_z)L_z}{l_{\text{veh}}} \tag{23}$$

$$S_z = 1800\cdot(l_z-y_z)(\text{veh/h}) \tag{24}$$

$$\rho_z(k) = \max\left(\rho_z(k-1),\ k-\text{ceil}\left(\frac{(c_z-w_z(k))l_{\text{veh}}}{(l_z-y_z)v_{\text{ff}}T}\right)\right),\ \rho_z(0)=0 \tag{25}$$

$$v_z^{c}(k) = \min\left(v_{\text{ff}},\ \frac{\sum_{j=k-t_w+1}^{k}u_z(j)L_z}{\sum_{j=k-t_w+1}^{k}x_z(j)t_w}\right),\ \forall z\in Z,\ k=t_w,\dots,K \tag{26}$$

$$w_z(k), m_z(k), a_z(k) \geqslant 0, \forall z \in Z, k = 1, \dots, K \tag{27}$$

$$y_z \in \{0, 1\}, \forall z \in Z_f \subseteq Z \tag{28}$$

$$y_z = 0, \forall z \notin Z_f \tag{29}$$

given:

$$m_z(0), w_z(0), x_{VQz}(0), v_z^c(0), d_z(k), t_{zi}(k), t_{z_0}(k), \eta_{zj}(k), \tag{30}$$

$$P_z^l(k), \beta_z^l, \delta_z^l, f_l(k) \tag{31}$$

$$\forall z \in Z, \forall \{(z,i) \in Z | z \in U_i\}, \quad \forall l \in B, k = 1, \dots, K$$

Vector $\mathbf{y}$ contains the decision variables of the problem, i.e. all binary variables $y_z, \forall z \in Z_f \subseteq Z$, indicating the presence (or not) of bus-only lanes inside every link z belonging to the set of candidate links Z_f. The first term of the objective function (Eq. (21)) refers to total PHT of bus passengers (see Eqs. (15)–(17)), the second term refers to PHT of car passengers (see Eq. (14)) and the third term represents the penalty cost associated with the total DBL operation and maintenance, which is proportional to total DBL length. In the cost term, γ denotes the cost per unit length of DBL added to the solution.The objective function has units of time, therefore cost γ is also expressed in units of time per DBL unit length.

Once a candidate solution is defined, the system performance is evaluated based on the evolution of queues in the network, as estimated by Eqs. (1)–(12) of the urban traffic model. The dynamic equations of the traffic model become constraints for the optimization problem, with proper integration of the decision variables $y_z, \forall z \in Z_f \subseteq Z$, as shown in Eqs. (22)–(25). Eq. (22) refers to saturation flow of approach $z-i$, with $z \in U_i$, where rl_{zi} is a binary indicator of whether the right-most lane in upstream link z is included in the set of l_{zi} lanes that allow movement to downstream link i. Eqs. (23)–(25) are simply Eqs. (6), (8) and (11), where l_z is replaced by $l_z - y_z$. Constraint in Eq. (27) ensures that all queues and arriving flows are non-negative, which is something that is also guaranteed by the traffic model itself. Eq. (28) defines decision variables to be binary and Eq. (29) guarantees that all non-candidate links of set Z_f do not get DBL. All necessary inputs such as initial system state, dynamic car demand in all origins, time-dependent turn and exit ratios, and traffic signal plans for all signalized intersections are listed in (30) and considered known. Total PHT estimation, according to Eq. (21) requires knowledge of time-varying bus occupancy and operational characteristics of all bus lines (see (31)). In the case where a target or upper limit of total DBL length exists, instead of specifying a non-zero cost γ, constraint (32) can be considered

$$\left| \sum_{z \in Z} y_z L_z - LT \right| \leqslant \theta, \tag{32}$$

where LT denotes the total target length and θ the acceptable deviation from that target. Finally, for any candidate solution, an iterative process of mode share adjustment and traffic simulation is performed, until mode shares estimated by Logit model based on simulation-generated travel times are close enough to those that were taken as inputs for the traffic simulation (see Fig. 1).

4. Solution methods

The optimal DBL allocation problem that is presented in Section 3, is a non-linear combinatorial optimization problem with binary decision variables and a finite feasible set that grows exponentially with the network size. In fact, the solution space size, if no constraint is imposed for total solution length, is $2^{|Z_f|}$, where $|Z_f|$ is the cardinality of the DBL candidate links set Z_f. Hence, for real instances, complete enumeration is not possible with current computing technology. Due to increased complexity and size, heuristic and metaheuristic algorithms seem to be the most promising tools for finding good quality solutions in reasonable time.

In this work, we construct and test a set of algorithms based on local search and Large Neighborhood Search (LNS), while integrating performance-improving techniques to guide the search process towards promising areas of the solution space. Firstly, we propose an algorithm to construct a good-quality DBL plan by recursively adding DBL at one road per step, after trying all currently available roads one-by-one and choosing the one leading to highest performance improvement. During this process, the algorithm can also be used to calculate performance indicators, or "scores", for all candidate links, according to the achieved system performance every time DBL placement is tested on each link. These scores can then be used in an LNS framework to drive the search towards potentially high performing solutions. Finally, a network decomposition technique is proposed as a complementary element of the repair process of LNS algorithm, with the aim of increasing search efficiency in cases of very large networks. The detailed description of the proposed algorithmic scheme is given in this section.

4.1. *Algorithm 1: Link-by-link plan construction and link score calculation*

A heuristic algorithm that constructs a good quality DBL plan, built on the principle of greedy, link-by-link addition, is formulated as shown in pseudo-code format, in Algorithm 1. The algorithm starts by setting as current solution the one corresponding to no DBL in the network, i.e. $y_z = 0, \forall z \in Z$ (Line 1); the objective function value for the current solution is calculated and stored (Line 2). Following a recursive process, one DBL per step is added to the solution, after evaluating, one by one through distinct simulations, all candidate links currently available (Lines 6–10). The objective function values (i.e. cost) corresponding to all potential additions are stored in vector $\boldsymbol{n}$. The best DBL addition of the current step is to link z^*, which results in the lowest overall cost compared to all other possible candidate link additions. If this cost is lower than the cost of the current solution (Line 13), the DBL is added to z^*, the current solution and the corresponding cost are updated (Lines 14–15), and the process is repeated until no further improvement is possible, i.e. none of the available candidate links can result in lower cost through DBL addition. Lines 17 and 19 relate to score calculation, are optional for Algorithm 1 and will be explained below.

Algorithm 1. Evaluation-based link-by-link plan construction and link score calculation

```
Data: initial solution $\mathbf{y}_0 = \mathbf{0}$   $(y_z = 0 \quad \forall z \in Z_f)$
Result: DBL plan $\mathbf{y}$, scores of candidate links $\omega$
1  $\mathbf{y} = \mathbf{y}_0$
2  $c = f(\mathbf{y}_0)$                                        $f$: objective function
3  $i^* = 0^+$
4  while $i^* > 0$ do
5      initialize $\mathbf{n} : n(z) = c, \forall z \in Z$
6      for $z \in \{Z_f | y_z = 0\}$ do
7          add trial DBL in $z$: $y_z = 1$
8          calculate and store new obj. function value $n(z) = f(\mathbf{y})$
9          remove trial DBL from $z$: $y_z = 0$
10     end
11     find best PHT improvement of step: $i^* = c - \min(n)$
12     store the link of best improvement: $z^* = \operatorname{argmin}(n)$
13     if $i^* > 0$ then
14         place DBL in link $z^*$: $y_{z^*} = 1$
15         c = f($\mathbf{y}$)
16     end
17     rank links according to increasing $n$ and store ranking $\mathbf{r}$
18 end
19 assign scores $\omega$ to links based on average overall ranking $\mathbf{r}$
20 return $\mathbf{y}$, $\omega$
```

Apart from constructing an improved solution from scratch, Algorithm 1 also serves the purpose of statistically assessing the potential of every link in improving system performance, if added to the DBL plan. In every iteration, the system performance resulting from all trials of DBL link addition is stored. At the end of every iteration, all links currently available for DBL addition are ranked in an increasing order of overall cost (Line 17). When no more improvement can be achieved by DBL addition, the algorithm stops and returns the current DBL plan $\mathbf{y}$ and a vector of rankings $\mathbf{r}$ per candidate link $z \in Z_f$, for every iteration of Algorithm 1. These scores represent the reported performance of the link, when added to the DBL plan, according to the solution construction process of Algorithm 1. These values can later be integrated in the search process of any general metaheuristic, in order to drive the search towards solutions with higher improvement potential. Score ω_z of every candidate link z ranges in $(0,1)$ and is calculated based on the following formula:

$$\omega_z = \max\left(\omega_{\min}, \frac{1}{|I^*|}\sum_{i \in I^*} \frac{N_f(i) - r_z(i)}{N_f(i)}\right) \tag{33}$$

In Eq. (33), ω_z denotes the score of link z, I^* the set of algorithm iterations during which link z is added in the trial solution, $N_f(i)$ the number of available candidate links at iteration i, i.e. size of set $\{z | z \in Z_f \text{ and } y_z = 0\}$, $r_z(i)$ the ranking of link z among all available links $N_f(i)$ at iteration i in increasing solution cost order ($r_z(i) = 1$ for link z, which leads to the lowest cost $f(y)$ if added to the DBL plan at iteration i), and $\omega_{\min} > 0$ the minimum score value that ensures a non-zero probability of selection. In summary, link scores ω_z, defined for every candidate link $z \in Z_f$, express the potential of the link to improve system performance by receiving a DBL, based on the assumption that links are included in the DBL plan in sequence of their ability to improve system performance (according to Algorithm 1). Link scores are used to drive the search process of the LNS algorithm described in the following section.

4.2. Large Neighborhood Search

The typical LNS algorithm, first proposed by Shaw (1998) with application to Vehicle Routing Problems (VRP), explores a large solution space by recursively "destroying" and "repairing" an incumbent solution in an effort to reach solutions of improved quality, i.e. lower objective function values (cost). The processes of destroying and repairing a current solution, initially defined for the VRP modeling structure, aim at resetting some of the decision variables of the solution and redefining them based on intuitive methods that can potentially lead to improved cost. The degree of destruction, referring to the number of decision variables that are reset in every iteration, is often large, thus allowing the algorithm to explore larger areas of the solution space and decreasing the chances of the algorithm getting trapped in local optima. If the newly constructed solution has lower cost with respect to the current one, it is accepted and replaces the current solution. Otherwise, it can either be immediately disregarded or, in order to diversify the search, it can be accepted with a probability depending on several criteria (as in Simulated Annealing, or SA). The stopping criterion can be based on a predefined number of iterations or execution time or it can depend on the current search performance. A detailed description of LNS metaheuristic as well as a review of its various applications in different fields and types of optimization problems can be found in Pisinger and Ropke (2019).

4.2.1. *Algorithm 2: Main LNS algorithm*

Algorithm 2. LNS for DBL allocation problem

```
Data: feasible initial solution $\mathbf{y}_0$, link scores $\omega$
Result: DBL plan $\mathbf{y}$
1  $s = \mathbf{y}_0$
2  $c = f(\mathbf{y}_0)$                                   $f$: objective function
3  for $iter$ = 1 to $iterMax$ do
4      $s' = \mathrm{destroy}(s, dd_{\max}, \omega)$
5      $s'' = \mathrm{repair}(s', rd_{\max}, \omega)$
6      $c' = f(s'')$
7      if $c' < c$ then
8          $s = s''$
9          $c = c'$
10     end
11     * update link scores every $\kappa_{\mathrm{int}}$ iterations          * optional
12 end
13 return $s$
```

The base structure of the proposed LNS framework is presented in Algorithm 2. An initial feasible solution, given to the algorithm as input, is set as current solution s (Line 1). This solution can be either randomly constructed or generated according to intuitive traffic engineering principles. In case there is a target length constraint, i.e. Eq. (32) applies, the initial solution is constructed accordingly. Current solution s is evaluated and its cost is stored as current cost c (Line 2). The iterative destroy and repair process follows (Lines 4–6). The maximum possible destruction and repair degrees, $dd_{\max}$ and $rd_{\max}$ respectively, are set and given as inputs to the destruction and repair processes, together with a set of link scores ω, which take values in the interval $(0, 1)$ and attempt to measure the efficiency, in terms of system performance, of a possible DBL installation in the link. These values can be generated by Algorithm 1, or defined differently, as it will be discussed later on. The destruction process modifies the current solution s by removing a set of DBL links. The result is the destructed solution $s^{'}$. Afterwards, the repair process receives $s^{'}$ and reconstructs it by adding a set of DBL links, while respecting all feasibility constraints. The newly formed solution, s'', is evaluated (Line 6) and its cost $c^{'}$ is compared to the current solution cost c (Line 7). Solution s'' replaces s as the current solution only if it leads to lower cost (Lines 8–9). While a different acceptance criterion can be used (e.g. as in SA), we choose a greedy approach, as we observed from a preliminary analysis, that in this case, the necessary flexibility in the search is sufficiently guaranteed and controlled by the destruction and repair degree of LNS mechanism and an SA acceptance criterion could decelerate the search (as more LNS steps are necessary), without achieving significant improvement in the final solution. Line 11 describes an optional process of link score updating, according to observed link performance in regular step intervals of LNS algorithm, that will be further discussed in a following section. LNS iterative process is repeated for a predefined number of times *iterMax*. After the last iteration, current solution s is the best found solution, which the algorithm returns as output to the user.

4.2.2. *Destruction process*

Typically, in LNS and A-LNS algorithms (see Ropke and Pisinger, 2006), the incumbent solution is destroyed and repaired through application of one or several specific methods that are expected to improve overall solution quality, according to the characteristics of the problem at hand. In the present approach, given the difficulty of identifying a deterministic way to describe correlations between DBL topology and the several elements influencing solution cost, such as queue spillbacks, resulting mode shares, or network traffic flows, we adopt a different approach. We utilize a system of link scores (weights) that describe the potential of every link to improve system performance by DBL addition. These scores are used as link selection probabilities, for DBL addition/removal during the destroy/repair processes for every iteration. The score values can be defined by simulation experiments, by prior execution of Algorithm 1, or based on a set of link characteristics that are intuitively connected to performance improvement in DBL presence (e.g. bus frequency per link), as we further discuss in a following section.

Algorithm 3. Destroy current solution s (Module for Algorithm 2, Line 4)

```
   Data: current DBL plan $s$, $dd_{\max}, \omega_z$
   Result: destroyed solution $s'$
1  calculate no of links to remove $d_{iter}$
2  $s' = s$
3  $counter = 0$
4  while $counter \leq d_{iter}$ do
5      specify current set of links with DBL
6      calculate removal probabilities: $p_z^r\ \forall z$
7      draw one link $l$ from the distribution of $p_z^r$
8      update solution $s'$ by removing link $l$: $y_l = 0$
9      $counter = counter + 1$
10 end
11 return $s'$
```

The destruction process of every LNS iteration (Line 4 of Algorithm 2), which consists of removing a number of DBLs from their current positions, is described by Algorithm 3. Given a maximum allowed destruction degree, $dd_{\max}$, set by the user, the actual destruction degree of every iteration is randomly drawn from an interval $(0,\ dd_{\max}(iter))$ with a uniform probability, where $dd_{\max}(iter) \in (0,1)$, takes the user-defined maximum value $dd_{\max}$ at the first iteration and then decreases linearly with the number of iterations, i.e. $dd_{\max}(iter) = dd_{\max}\left(1 - \frac{iter-1}{iterMax}\right)$. This is done in order to allow larger possible search steps at the first iterations of LNS algorithm, which increase the chances of identifying promising areas of the solution space by escaping local optima, and smaller steps in the last iterations, in order to fine-tune the solution through local search. The actual number of links to be removed at iteration $iter$, denoted as d_{iter}, is specified (Line 1) by the following formula:

$$d_{iter} = \text{ceil}\left(r \cdot dd_{\max} \cdot \left(1 - \frac{iter-1}{iterMax}\right) \cdot \sum_{z \in Z_f} y_z\right) \tag{34}$$

In (34), r is a random number drawn from a uniform distribution $\mathscr{U}(0,1)$. Operator `ceil` ensures that an integer number of links greater than or equal to one will be removed from the solution in every iteration, provided that there is at least one link in the current DBL plan, i.e., $\sum_{z \in Z_f} y_z \geqslant 1$.

Links are then removed one-by-one, in an iterative process. A set of link scores is used to define selection probabilities for removal from the current DBL plan. Since link scores express the links' estimated potential in improving system performance by DBL addition, we simply use the difference of every link score ω_z from 1 to define link selection probability for removal. Therefore, in every step, one link z is selected for removal from the set of links currently having DBL, with probability

$$p_z^r = \frac{1 - \omega_z}{\sum\limits_{z \in \{Z_f | y_z = 1\}} (1 - \omega_z)} \tag{35}$$

where, p_z^r denotes the probability of link z to be selected for removal and ω_z is the link score calculated for link z (e.g. provided by Algorithm 1 according to formula (33)). Obviously, the probability of any link z to be selected for removal is higher if its score ω_z is lower compared to the scores of the rest of the links currently in the solution. After completing the draw from the distribution of p_z^r, with $z \in \{Z_f | y_z = 1\}$, the DBL is removed from the chosen link and the solution is updated. This process, described in lines 5–9 of Algorithm 3, is repeated until all d_{iter} links have been removed. Then the destructed solution s' is returned as output and the process of Algorithm 2 continues to the repair process of line 6.

4.2.3. Repair process

Algorithm 4. Repair destroyed solution $s^{'}$ (Module for Algorithm 2, Line 5)

```
   Data: destroyed DBL plan $s'$, $rd_{\max}$, $\omega_z$
   Result: repaired solution $s''$
 1 $^\star$calculate no of links to add $r_{iter}$                $^\star$ when no length constraint applies
 2 $s'' = s'$
 3 $counter = 0$
 4 while $counter \leq r_{iter}$ do
 5     specify current set of links with no DBL
 6     calculate addition probabilities: $p_z^a$
 7     draw one link $l$ from the distribution of $p_z^a$
 8     update solution $s''$ by adding link $l$: $y_l = 1$
 9     $counter = counter + 1$
10 end
11 return $s''$
```

The repair process varies depending on the validity of constraint (32) about total DBL length. We examine the case where no such constraint applies. The addition process is described in Algorithm 4, which receives as input the destroyed solution $s^{'}$ (output of Algorithm 3), the user-defined, maximum possible repair degree $rd_{\max}$, and link scores ω_z. The number r_{iter} of links to be added in the destroyed DBL plan $s^{'}$ during the repair process is defined by formula (36) below, similar to (34).

$$r_{iter} = \text{ceil}\left(r \cdot rd_{\max} \cdot \left(1 - \frac{iter - 1}{iterMax}\right) \cdot \sum_{z \in Z_f} (1 - y_z)\right) \tag{36}$$

Links are added to the destroyed solution one-by-one, in an iterative process until the required number of additions, r_{iter}, is reached. In every iteration, the set of links currently available (i.e. $z \in Z_f$ with $y_z = 0$) is specified. One link z is drawn from this set with a probability

$$p_z^a = \frac{\omega_z}{\sum\limits_{z \in \{Z_f | y_z = 0\}} \omega_z} \tag{37}$$

In (37), p_z^a denotes the probability of link z to be selected for addition in the current step. After the draw from the distribution of p_z^a is complete, the selected link is added to the destroyed solution $s^{'}$. The same process (Lines 5–9) is repeated until all r_{iter} DBL links are added to the solution. The repaired solution, s'' which is the output that Algorithm 4 returns in the main LNS Algorithm 2 (Line 5), consists the newly formed solution of the current LNS iteration. It will be evaluated through the process depicted in Fig. 1 and it will replace the incumbent solution s only if it results in better system performance.

The probabilistic selection of links to be added and removed in every LNS iteration ensures the necessary diversification in the search process that helps the algorithm avoid getting trapped in local optima. In case where a length constraint such as (32) applies, the repair process is performed slightly differently. No repair degree is considered, since the number of links to be added back in the solution will depend on the current DBL solution length. Link addition is done again one-by-one in an iterative way but total solution length is calculated after every single link addition. The process terminates when the solution length reaches the desired value, so that length constraint (32) is satisfied. The selection process for link additions remains the same as shown in lines 5–8 of Algorithm 4.

4.2.4. Alternative link scores and score updates

Acquiring link score values ω_z for all candidate links, in order to be used in LNS processes can be done by running Algorithm 1, by integrating the score updating process of lines 17 and 19. Score values acquired likewise are expected to be highly effective in driving the search process of LNS algorithm towards promising solution space areas, because they are specified by trial-and-evaluation, which considers all possible effects of DBL setting on system characteristics, such as mode shift, queue spillbacks, bus passengers delay savings in the respective links, etc. Moreover, the solution building process of Algorithm 1 by definition leads to improved solutions in every step. However, this is a computationally expensive process, due to the large number of trials and simulation runs that Algorithm 1 needs to perform in order to identify the best link addition in every step and estimate the respective link scores.

Therefore, there might be cases where such a computationally expensive process cannot be afforded, for example when several different values of the initial inputs (e.g., dynamic travel demand profile, car trip routing patterns, bus operational characteristics, etc.)

must be considered. For such cases, we propose the following alternative approaches:

- Utilize link scores according to link characteristics that are expected to lead to good quality solutions, e.g., bus frequencies, link bus passenger flow, road space availability, etc.
- Utilize uniform or other type of link scores (e.g., based on bus frequencies) for all candidate links and include a score update process in regular intervals in LNS algorithm (Line 11). Link scores that are used for removal and addition are updated, after a specific number of LNS iterations, according to the observed performance of the respective solutions. In this way, link scores are gradually adjusted to the specific case study during LNS execution, in a similar way as A-LNS metaheuristic evaluates and updates scores of different destroy and repair methods based on their performance (for details see Ropke and Pisinger, 2006).

It should also be noted that these options can be combined, e.g., if the update step of line 11 is included in LNS algorithm, the initial scores can either be uniform or based on current bus frequencies or even come from Algorithm 1 executed with different input data. However, LNS algorithm may require significantly more iterations in order to reach good quality solutions, as the learning process required for the proper tuning of scores is built through iterations. Score updates can be performed in regular step intervals according to the following equations:

$$\omega_z(\kappa+1) = \lambda\omega_z(\kappa+1-\kappa_{\text{int}}) + (1-\lambda)\left(\delta_z^a(\kappa) + \left(1-\delta_z^r(\kappa)\right)\right) \tag{38}$$

$$\delta_z^r(\kappa) = \frac{\sum_{i=\kappa-\kappa_{\text{int}}+1}^{\kappa} \beta_z^r(i)\frac{c_i-c_i'}{c_i}}{\sum_{i=\kappa-\kappa_{\text{int}}+1}^{\kappa} \beta_z^r(i)} \tag{39}$$

$$\delta_z^a(\kappa) = \frac{\sum_{i=\kappa-\kappa_{\text{int}}+1}^{\kappa} \beta_z^a(i)\frac{c_i-c_i'}{c_i}}{\sum_{i=\kappa-\kappa_{\text{int}}+1}^{\kappa} \beta_z^a(i)}. \tag{40}$$

Eq. (38) is used to update scores ω_z for all candidate links $z \in Z_f$, every κ_{int} iterations of LNS algorithm, according to the performance of the solutions tested in these κ_{int} steps and formed by adding or removing link z. In this equation, κ denotes the LNS iteration at the end of which scores are updated, and $\kappa-\kappa_{\text{int}}+1$ the iteration before which scores were last updated; λ is the decay factor, which is user-defined and dictates how much the new score values will be influenced by their previous value; $\delta_z^r(k)$ and $\delta_z^a(k)$ represent the score update terms, which are based on the average performance of all trial solutions that were formed by removing and adding link z, respectively, during iterations $\kappa-\kappa_{\text{int}}+1$ to κ. Their calculation is derived according to Eqs. (39) and (40), where i is the index for LNS iterations; $\beta_z^r(i)$ is a binary indicator that is equal to 1 if link z is removed from the current solution at iteration i, and 0 otherwise; $\beta_z^a(i)$ is a similar binary indicator about link addition; c_i is the cost of current solution s at iteration i; c_i' is the cost of newly formed solution s'', after completion of destroy and repair processes of iteration i. In other words, terms $\delta_z^r(k)$ and $\delta_z^a(k)$ express the average relative cost change every time link z was removed from and added to the newly formed solution, respectively, during the last κ_{int} LNS iterations. Note that since link scores, by definition, express the potential of links to improve solution by DBL addition, δ_z^r is subtracted from 1 in Eq. (38), in order to translate the potential of removal to potential of addition. The relative change of cost, calculated by fraction $(c_i-c_i')/c_i$, can also be normalized by the highest absolute value of relative cost change of this set of κ_{int} iterations.

4.2.5. *Alternative repair process using sub-networks*

A complementary element that can be integrated in LNS repair process is proposed and evaluated as part of this algorithmic scheme, intended especially for application in large network instances. With the aim of increasing LNS algorithm's chances of finding good quality solutions, the repair process of destructed solutions is enhanced by including a trial and evaluation step including smaller networks, in the proximity of candidate links, which we call sub-networks. More specifically, the main LNS algorithm remains as is (Algorithm 2), and so does the destruction process (Algorithm 3), while the repair process is modified as shown in Algorithm 5 below. As described in lines 8–14, every DBL addition is made at the best performing link, out of a sample L of available links selected according to their scores, after evaluating the traffic performance of isolated sub-networks around them. By isolating a sub-network around a candidate link, a set of before-after DBL scenario evaluations, that last for a fraction of total simulation time, can give a good estimation about the potential performance of DBL in this link, with much lower computational cost compared to a full-network and full-time simulation process. The difference of this enhanced repair module with respect to the process described in Algorithm 4 is that link scores are iteratively used to create sample sets of candidate links for DBL addition. Before every DBL addition, for every link of the sample set, the algorithm evaluates the performance of the respective sub-network with and without DBL (lines 10–11). Then, actual DBL addition is done to the best performing link of the set (lines 14–15). This process is repeated for every DBL addition with a new sample set of available links created each time. When all additions of the step are made, the repair process is completed and the newly formed solution s'' is returned to the main Algorithm 2 and evaluated with a full-network full-time simulation.

Algorithm 5. Repair the destroyed solution $s^{'}$ with sub-network evaluations (Alternative module for Algorithm 2, Line 5)

```
Data: destroyed DBL plan $s'$, $rd_{\max}$, $\omega_z$
Result: repaired solution $s''$
1  * calculate no of links to add $r_{iter}$                    * when no length constraint applies
2  $s'' = s'$
3  $counter = 0$
4  while $counter \leq r_{iter}$ do
5      specify current set of links with no DBL
6      calculate addition probabilities: $p_z^a$
7      draw a sample set of links $L$ from the distribution of $p_z^a$
8      for $j \in L$ do
9          create sub-network around link $j$
10         evaluate performance of sub-network without DBL in $j$, $f_0$
11         evaluate performance of sub-network with DBL in $j$, $f_1$
12         store improvement $f_{sbn}(j) = f_0 - f_1$
13     end
14     find best link $l \in L$, so that $f_{sbn}(l) < f_{sbn}(j), \forall j \in L, j \neq l$
15     update solution $s''$ by adding link $l$: $y_l = 1$
16     $counter = counter + 1$
17 end
18 return $s''$
```

The sub-network surrounding any link of interest is composed by a set of interconnected links upstream and downstream of the central link of interest, in the form of tree branches, whose distance from the central link, measured in number of consecutive links, is less than a predefined number, which dictates the sub-network size. This set of links together with their start and end nodes form the basis of a sub-network. To maintain local simulation accuracy, links upstream/downstream of the central link that are not directly connected to it but receive (or send) flow from (to) other sub-network links, are also included in the sub-network. An example of different size sub-networks around the same link of interest can be seen in Fig. 2. Sub-network size, as well as the considered traffic simulation time window, affect the accuracy of the assessment of the potential impact of DBL in central link. A smaller size sub-network and a short selected simulation period might lead to misleading results, which would not be verified by a full-network full-time simulation test. However, smaller sub-network size and simulation time result in lower computational cost.

Traffic simulation in sub-networks is done in the same way as in full network, by using the same traffic model. Among sub-network links, those with no upstream link function as origin links, while links with no downstream link function as destination links. By using the dynamic link inflows of the sub-networks' origin links, that were calculated by the latest full-network simulation, as input demand for the sub-network, as well as the turn and exit ratios of the full-network, we simulate traffic in every sub-network with and without considering DBL presence in the central link, for a short period of peak hour. The initial sub-network traffic state is the same as in the last full-network simulation, at the time step corresponding to the start of the sub-network simulation. Also, DBLs placed in other sub-network links apart from the central, according to destroyed solution $s^{'}$, are always considered in sub-network simulations and their position does not change. Only central links are tested with and without DBL. PHT difference between the two scenarios (with and without DBL) is used as the evaluation criterion of the considered DBL addition. However, the accuracy of the sub-network evaluations is expected to be reduced in iterations with high destruction or/and repair degree, as DBL network configuration differs significantly at the beginning and end of the repair process, meaning that the first sub-network evaluations are done in significantly different surrounding conditions compared to the last ones. This is the reason that this method proves more efficient with smaller scale solution modifications, similar to the effectiveness of LS techniques.

4.3. *Summary of the solution scheme*

The proposed solution scheme that is described in details in Section 4, is graphically represented in Fig. 3, where the connections between the algorithmic components discussed above, as well as the sequence of the proposed solution's procedure are depicted. Dashed lines indicate alternative options. For instance, improved (optimized) solutions can be produced either by Algorithm 1 or 2, executed separately, or by executing first 1 and then 2, where outputs of the first one can be used as inputs in the second one (e.g. initial solution or simulation-based link scores). As we will see in the following section, Algorithm 1 performs significantly higher number of solution evaluations that require longer execution time, for the generation of a single improved solution. However, at the same time, it can generate a set of simulation-based link scores, that are highly effective in indicating good candidate links for DBL introduction and can be used in LNS algorithm (Algorithm 2), to increase its efficiency. Nevertheless, the latter can be executed on its own without prior execution of Algorithm 1, by using link scores that are either uniform or generated based on bus operational characteristics, with the possibility of integrating an update process that adjusts link scores based on their performance in the process of LNS. The repair process

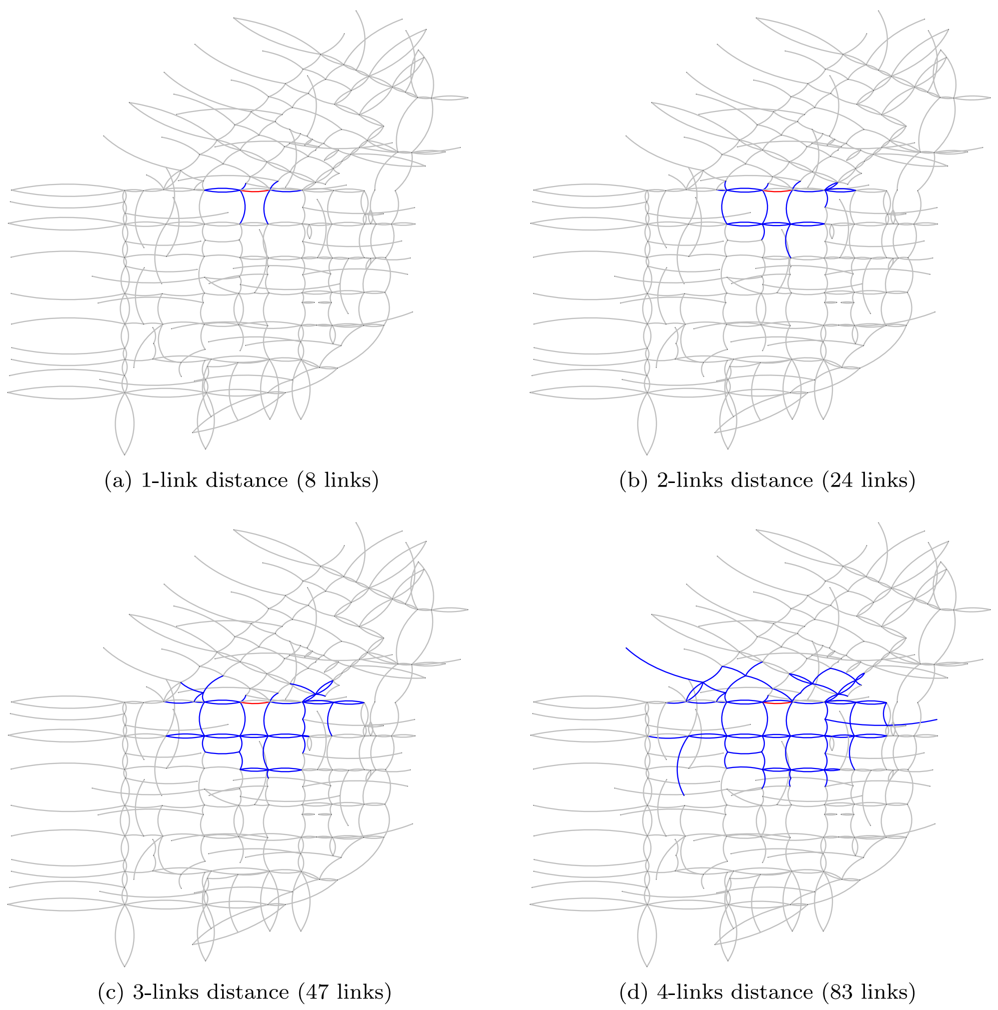

(a) 1-link distance (8 links)
(b) 2-links distance (24 links)
(c) 3-links distance (47 links)
(d) 4-links distance (83 links)

Fig. 2. Sub-networks of different size in the neighborhood of the same reference link (in red). Distance refers to the number of consecutive and connected links included, upstream and downstream of the central link.

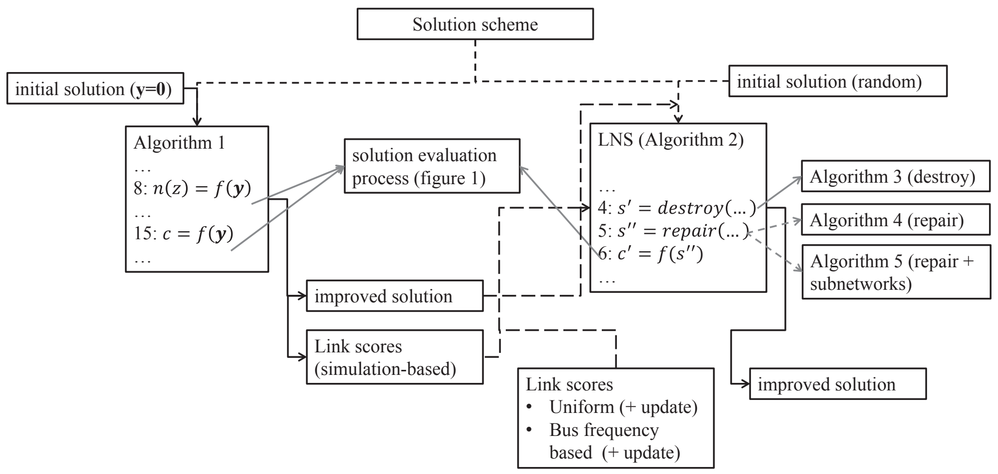


Fig. 3. Overview of the proposed solution scheme and the relation between algorithms, inputs and outputs; dashed lines indicate alternative options.

of LNS can be performed by either Algorithm 4 or 5. The latter performs sub-network evaluations to increase the possibility of building improving solutions, especially in very large networks. Finally, the evaluation of any newly formed solution follows the process depicted in Fig. 1.

5. Numerical application

5.1. Case study

The proposed method is applied to part of the traffic network of San-Francisco central area, in California, USA (see Fig. 4(a)). The network is composed of 426 links and 267 nodes, out of which 156 represent signalized intersections on a pre-timed traffic signal control plan with cycle lengths up to 100 sec. Out of all links, 96 are labelled as candidate for bus-lane setting (see Fig. 4(c)), corresponding to a total of 10.6 lane-kilometers. Candidate links are in principle all links that are included in the route of at least one bus line and have two or more lanes in total. In order to avoid excessive disturbance of car traffic, links with only two lanes were included in the candidate set only in case of considerably high bus frequencies.

The dynamic profile of car trip demand has a trapezoidal shape, as depicted in Fig. 5(a), where the total generating car flow in the network is shown over time; this demand is distributed in 42 origin nodes. There exist 29 bus lines travelling in the studied region. The distribution of bus routes in the network can be seen in Fig. 5(c). Bus frequencies and passenger ridership information are chosen so as to replicate realistic conditions: we assume 6 buses/hour (headway of 10 min) for all bus lines and bus occupancies that follow a trapezoidal profile over time, similar to the car demand profile. Average bus lines ridership in the central roads is assumed double than in peripheral roads. Bus average ridership is different for every bus line, ranging from 20 to 65 pax/bus during peak hour according to Fig. 5(b). Aggregated transit passenger flow during peak-hour is shown in Fig. 5(d). Initial shares of total demand per mode are assumed equal to 74% for cars and 26% for buses, while total number of commuters is equal to 87 K. Time-varying turn ratios, reflecting route choices of car users, are calculated by a preliminary microscopic simulation analysis performed with Aimsun software, for the case where no DBL is considered. Average turn ratios for every approach are calculated in periods of 15 min. Without loss of generality, turn ratios are considered unaffected by DBL setting in the present study, based on the assumption that bus-only lanes introduction will not significantly affect the route choices of drivers; this can be changed in the future and use the same approach with turn ratios that can be adjusted to the specific DBL configuration tested.

Every potential DBL plan is evaluated by a sequence of traffic simulations followed by mode choice adjustments until convergence is achieved (as shown in Fig. 1). Every traffic simulation is performed for a 10 h period, in time steps of 5 s, in order to guarantee that network is empty by the end of simulation time (in most cases network is empty after 4-5 h).It should be noted that in case where different DBL assignment for the evening peak is possible, the whole process needs to be repeated, by considering car/bus demand information of the evening period, in order to identify the best possible DBL assignment for the evening hours. In our case, we study only the morning peak period and derived solutions correspond to the morning DBL assignment. Average free-flow speed for cars and buses is assumed equal to 25 km/h. The same value is set for buses and cars for simplicity, since we observed in preliminary

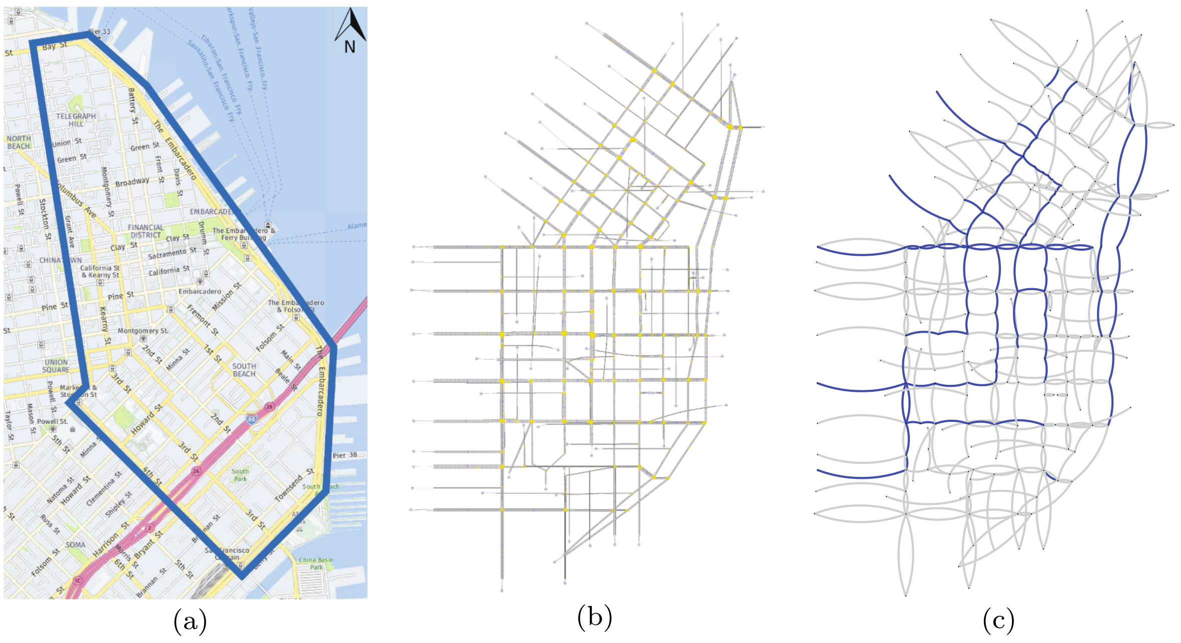
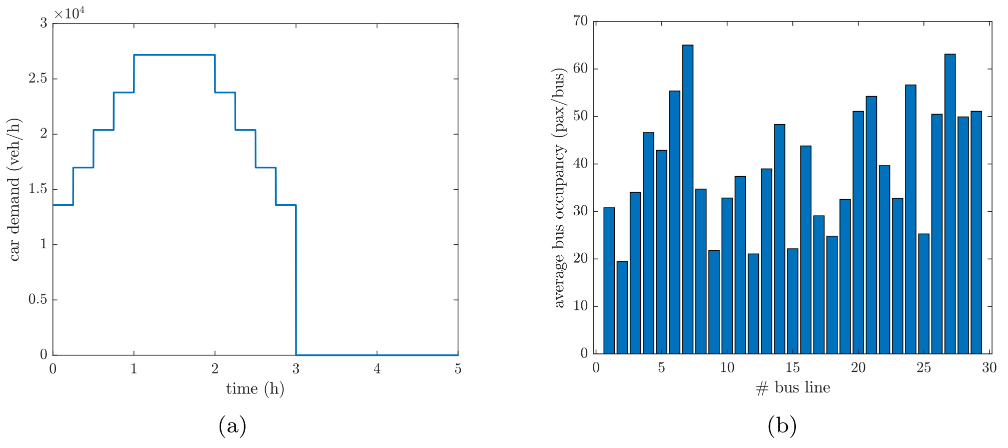


Fig. 4. The San Francisco road network; (a) Map of the studied area. (b) Model of the network in Aimsun. (c) Map of candidate roads for bus-only lane installation (in blue).



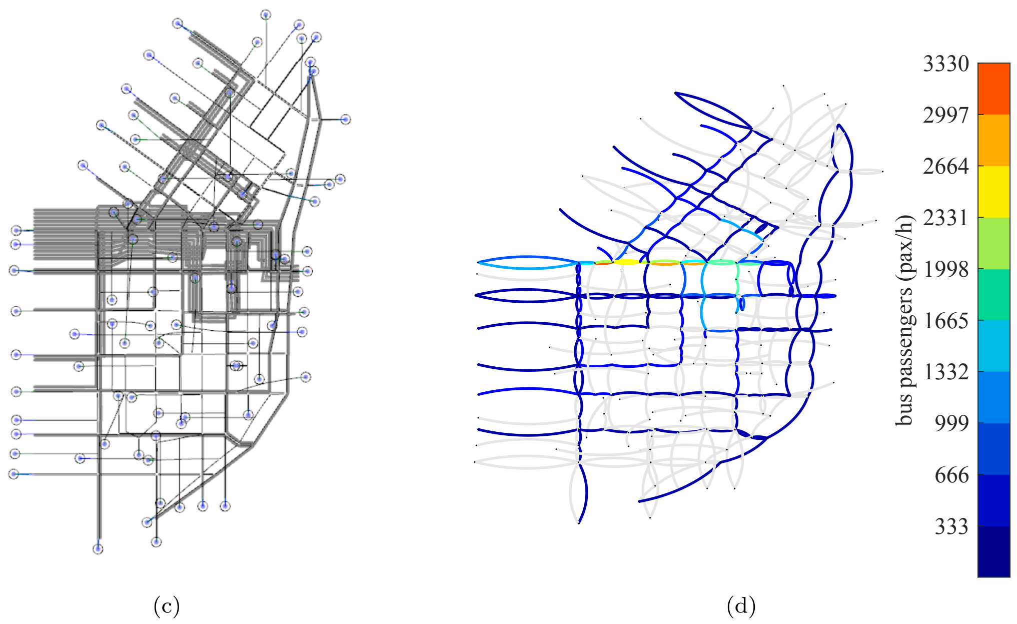


Fig. 5. Case study input data about car and bus travel demand; (a) Profile of the initial total car travel demand over time; (b) Average passenger ridership of buses per line in peak-hour; (c) Bus routes distribution in the network; (d) Average bus passenger flow per link in peak-hour.

experiments that the results are not sensitive to small variations of bus free-flow speed. Average vehicle length is assumed equal to 5 m and average car occupancy 1 pax/car. For dwel time estimation, we assume that at every bus stop, the average fraction of boarding and alighting passengers, β_z^l, of Eq. (17), is assumed to be roughly 30% of the current bus occupancy.

Logit model parameters are defined as follows: $ASC_c = 1.074$, $ASC_b = 0$, $\beta_c = -2.578$, $\beta_b = -9.294$. The iterative process of Fig. 1 is terminated if the percentage of passengers changing mode with respect to previous state is less than 0.1% of the total number of passengers or after 50 iterations at most, in case no convergence can be achieved.

5.2. Problem variants

The proposed algorithmic scheme is applied for the case study described in Section 5.1. Algorithm 1 which generates a step-by-step constructed solution and link scores estimation, is executed first. Then, LNS Algorithm 2 is executed using as link scores those estimated by Algorithm 1, in addition to other approximations (see below), in order to efficiently explore the solution space. The two algorithms are applied for the following variants of the DBL optimization problem:

1. Minimize total passenger travel time without penalty related to total solution length ($\gamma = 0$ in Eq. (21)) and with an additional constraint for total DBL length (Eq. (32)).

2. Minimize sum of total passenger travel time plus penalty cost for road space occupied by DBL (Eq. (21)) without additional constraint for total DBL length. Penalty cost per unit length of DBL installed is set to $\gamma = 750$ hours/(km day). We estimate this value by considering hourly total maintenance cost of DBL equal to 1715 USD/lane-kilometer, 10 hours of daily DBL operation and value of time equal to 22.90 USD/hour.

5.3. Results

5.3.1. Algorithm 1

Algorithm 1 is applied for both problem variants 1 and 2. However, link score calculation is only executed for variant 1, where only total PHT is included in the objective function. This is done so that scores reflect the potential of each link in improving system performance without considering their length. Algorithm 1 outputs include a good quality DBL plan and a set of link score values. Fig. 6 (a) shows the solution performance at the end of every iteration of the algorithm, for problem variant 1. As expected, total PHT is reduced in every step (with step 0 corresponding to no DBL scenario), after best link addition, following evaluation of all available options. The red line shows the car users percentage reduction, in response to improved bus travel time resulting from DBL setting, according to the utilized Logit model. For comparison reasons, the dashed line shows the system traffic performance for the initial network state (no DBL in the network), but considering the reduced car demand, according to the estimated mode shift of every step of Algorithm 1. The difference between dashed and solid blue lines indicates that a significant part of the network performance improvement, in presence of an efficient DBL plan, results exclusively from smart DBL distribution and is not related to the assumed mode shift from car to bus. In other words, even if we achieve mode shift from cars to buses by different means (dashed line), without DBL installation, the improvement of PHT is still smaller than in the presence of an efficient DBL plan (solid line). This can be explained by the fact that optimized DBL location selection can take advantage of unused road space surplus in specific roads and cause minimum car traffic disturbance. Another possible explanation is that for specific DBL configurations, the restricted car flow owing to the reduced general-purpose lanes can produce an effect of perimeter control. Limiting vehicle access to congested parts of the network can have a beneficial effect on vehicles, even if some of them are forced to wait longer in queues. This happens because the high-demand region remains below or close to capacity while, at the same time, bus delays are reduced due to DBL presence.

Fig. 6(b) shows PHT improvement in relation to the gradual increase of the total DBL lane-kilometers installed, as links are added in the DBL plan. The algorithm stops adding links when no further PHT improvement can be achieved. In the case of Fig. 6 (problem variant 1), the best solution found includes DBL in almost all candidate locations (a total of 9.1 lane-kilometers) mainly because of the continuing mode shift, given that no constraint for total length is imposed in this case. Link scores ω, representing the average ranking-based scores of all trials, generated by Algorithm 1, are shown in Fig. 7, with the whiskers' length representing the standard deviation over all link trials. Scores take values in the range (0,1), with larger scores indicating good performance of DBL if placed in the link. In the figure, links are sorted, from left to right, in the sequence they are added in the DBL plan. Links added earlier in the solution are tested fewer times and therefore have a smaller standard deviation. As expected, there is some correlation between average score values and sequence of addition, even though there are some low link scores with high standard deviation that are added in relatively early steps. However, it is expected that score values can increase the efficiency of LNS framework of Algorithm 2. Fig. 8 shows results of Algorithm 1 applied for problem variant 2. As expected, the algorithm stops adding DBLs to the solution much earlier and the final solution is more efficient, meaning that the ratio of achieved PHT improvement over DBL-occupied road space is considerably higher. While Algorithm 1, in principle, results in good quality solutions (as it involves enumeration of a large number of possible DBL additions per step) and its execution is required if we need to extract simulation-based link scores, it is highly expensive computationally

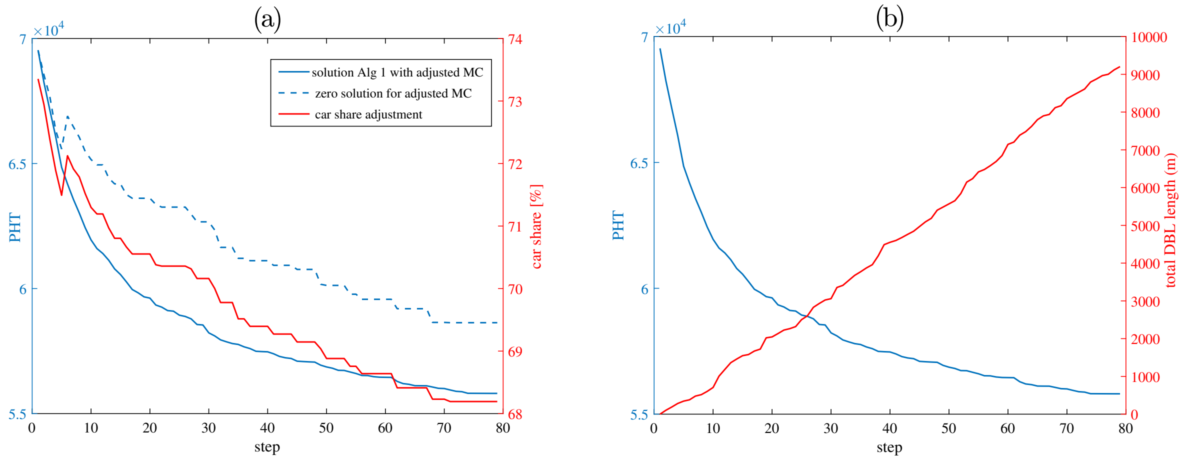


Fig. 6. Algorithm 1: Gradual improvement of solution for problem variant 1 (PHT Only); (a) Step-by-step decrease of PHT (blue solid) and respective car share adjustment (red) as DBL plan is constructed. In dashed line the PHT of the initial state (no DBL) considering the adjusted car share (red line); (b) Step-by-step PHT decrease vs DBL solution length (red).

(4506 and 2313 DBL plans tested for problem variants 1 and 2, in 25.39 h and 12.36 h, respectively), while there is no guarantee of optimal solution. This is why the optimization framework is enhanced by using LNS Algorithm 2, which can produce similar quality results with much fewer DBL plan evaluations (100 per replication in our case).

5.3.2. LNS algorithm

LNS, as described in Algprithm 2, is applied for problem variants 1 and 2. Since running Algorithm 1 to acquire link score values might not always be possible, due to its high computational cost, we test different performance indicators (scores) to estimate the potential of each candidate link in improving network performance if included in the DBL plan. In order to evaluate the role of link scores in algorithm performance, we execute LNS for problem variant 2 several times, using different sets of link score values for the destruction and repair of the solution, which we label as follows:

- *Uniform*: All link scores are equal. All links have the same probability to be chosen for removal or addition of DBL.
- *Bus Frequencies*: Every link score is equal to the ratio of total bus frequency of the link over the maximum bus frequency observed among candidate links.
- *Scores*: Link scores are found by Algorithm 1 based on simulation trials, according to Eq. (33).
- *Uniform + Update*: All link scores are initially equal but are being updated within LNS algorithm, according to the process described in Section 4.2.4.
- *Bus Frequencies + Update*: Links are initially assigned scores according to their bus frequencies (see above) but are being updated within LNS algorithm, according to the process described in Section 4.2.4.

For all cases, LNS algorithm is executed 10 times, for 100 steps each, starting from a random initial solution with total DBL length corresponding to around 50% of the total candidate space. The algorithm takes as input the same set of initial solutions for all above cases. Maximum destroy/repair degree for all cases is set to $dd_{\max} = rd_{\max} = 30\%$ of number of links and re-adding the same links at the same step is not allowed, in order to avoid circling around the same solutions. For the last two cases of the above list, where link scores are being updated within LNS, the algorithm performs 250 steps, in order to include a warm-up part. The decay factor is set to $\lambda = 0.5$ and the update is done every $\kappa_{\text{int}} = 10$ steps.

In Fig. 9 we show boxplots with distributions of total PHT with and without penalty cost and respective total DBL lane-kilometers of the initial and best found solution sets, after executing LNS algorithm for every score case listed above, for problem variant 2. In Fig. 9 (a) we observe that LNS is successful in improving a random initial solution of average quality, even by using uniform link scores for the destroy and repair processes, but the best performance is observed when link scores are computed by Algorithm 1. Small dispersion of the best found solutions for this case indicates higher probability of finding a good quality solution with fewer replications, compared to using other types of link scores. Link scores based on bus frequencies are also more efficient compared to uniform scores, showing, as expected, that higher bus frequency indicates more suitable link for DBL setting. Algorithm performance when less efficient link scores are used (such as uniform or bus-frequency-based), can be improved by applying the update process described in Section 4.2.4. However, a larger number of iterations is necessary for the update process to be effective. We can see from Fig. 9(a) that by updating link scores, best found solutions are improved with respect to cases without score update. This means that prior execution of Algorithm 1 is not necessary for efficient LNS application. The utility of an update process is more obvious in cases of different scenario evaluation, e.g., different demand or bus input data, where repeated execution of Algorithm 1 would be unrealistic. By comparison of Figs. 9(a) and (b) we can see that solving the problem variant 2, i.e., with penalty cost included in the objective function and no solution length constraint, the algorithm seeks space-efficient solutions, where a compromise between PHT improvement and road space given to DBL is made. This is obvious if we compare the characteristics of the best found solutions of all tests in Figs. 9(a)–

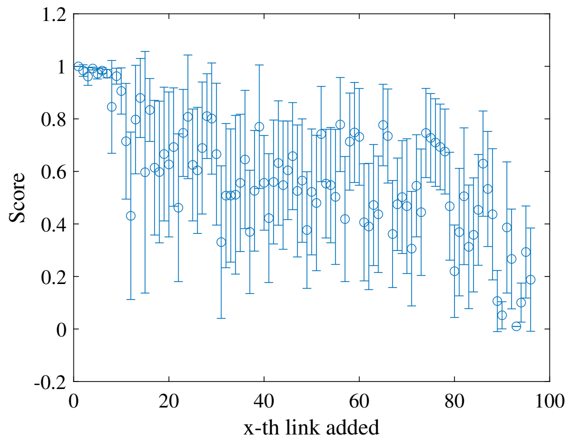


Fig. 7. Link scores calculated by Algorithm 1: Mean ± standard deviation of all trials for candidate links in the sequence they are added to the solution (left to right).

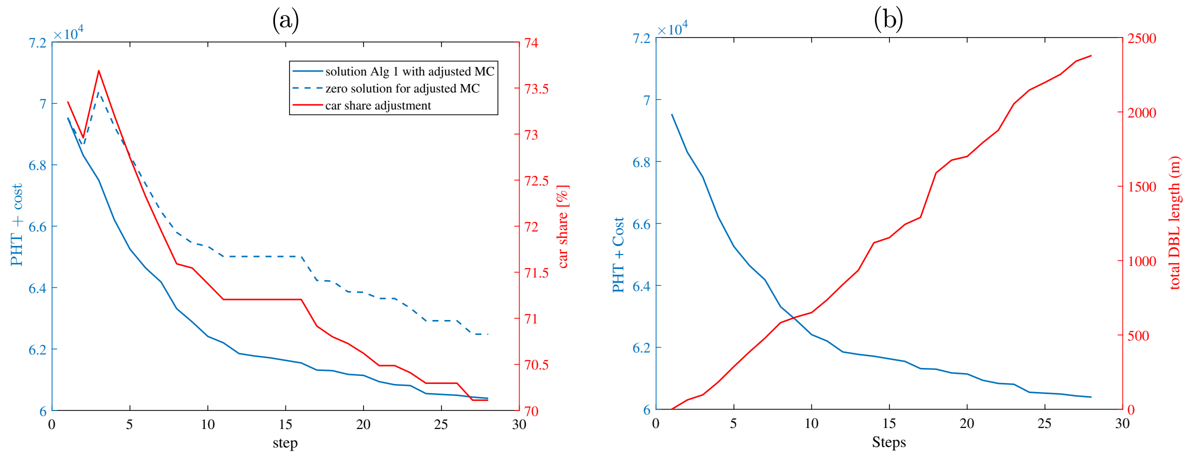


Fig. 8. Algorithm 1: Improvement of solution for problem variant 2 (PHT + Cost); (a) step-by-step decrease of the solution cost (blue solid) and the resulting car share (red) as DBL plan is constructed. In dashed line the cost of the initial state (no DBL) considering the car share of the red line; (b) step-by-step cost decrease and solution length (red) as DBL plan is constructed.

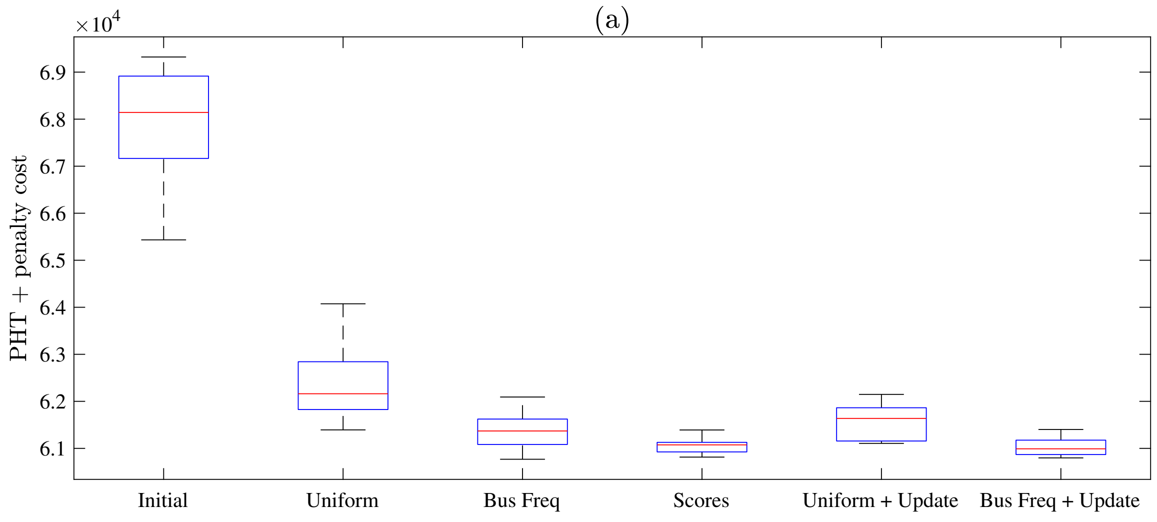
(c), where we observe that even though the solutions found by “Scores” or “Bus Frequency + Update” tests have lower values of combined PHT plus penalty cost, PHT improvement is smaller compared to solutions of other cases; however, they occupy less road space for DBL. Therefore, the solutions that we consider best in variant 2 are those having the largest PHT improvement per lane-km of DBL. We also note that total lane-kilometers of the initial random solutions is not binding, since LNS algorithm can increase or decrease total DBL plan length during the search process, due to the way destruction and repair are done and the greedy acceptance criterion for new solutions.

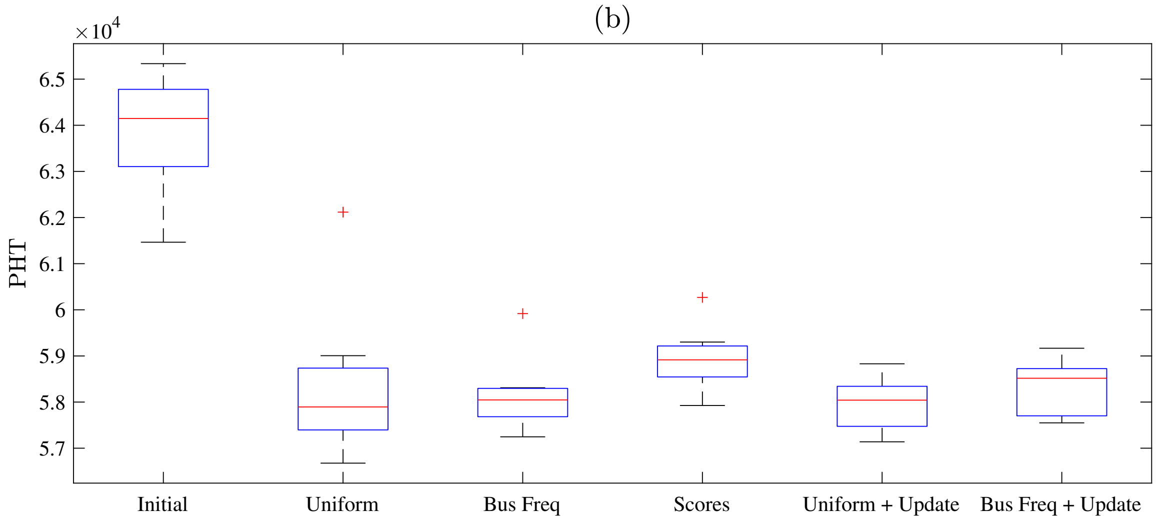
Effectiveness comparison between link score types used in LNS for problem variant 2 is complemented by Fig. 10, where one can see the evolution of LNS search for every link score type used. The graphs show in boxplots the cost of the best solution found so far after every LNS iteration, for all replications of every score case. It is evident that using simulation-based link scores of figure (c), coming from Algorithm 1, leads to better solutions in fewer LNS iterations compared to cases (a) and (b), while final solution costs of 10 replications show small variance, in contrast to the “Uniform” case, where costs of best found solutions after 100 steps vary significantly (larger IQR), indicating weaker optimization convergence (more steps necessary). The degree of similarity between best found DBL plans is shown in Fig. 11, where the observed frequency of every link’s appearance in the final DBL plan, after 10 LNS executions, is displayed. While the set of random initial solutions is composed of diverse solutions, the best solutions found by LNS by using any type of link scores show increased similarity. More specifically, several “good” DBL candidate links are identified and included in the final DBL plan, in most or even all replications, in more efficient score type cases, such as “Scores” and “Bus Frequency + Update”. On the contrary, not promising links are rarely or never included in the final solution.

LNS using the alternative repair process described by Algorithm 5, which is mainly proposed for very large network instances, is tested for our case study with a smaller destroy and repair degree, resulting in small changes of the incumbent solution per LNS step. The smaller modification degree is considered important for the effectiveness of this repair strategy, because accuracy of sub-network evaluation, dictating the next best position for DBL addition out of a stochastically created subset of available links, depends on the surrounding network conditions. Therefore, in a large destruction case, where many of DBL links have been removed from the incumbent plan, sub-network evaluations performed in the first repair iterations assume surrounding conditions (neighboring DBLs) significantly different than those towards the last repair steps, when most DBL links have already been re-added. However, in very large networks, the process of sub-network evaluation, driving the repair process of the solution, is expected to increase search efficiency, even by using small destruction and repair degree. In our case, we apply this alternative repair process in LNS algorithm for 10 replications of 200 steps each, by starting from the same set of random initial solutions as in previous cases, for problem variant 2, with maximum destroy and repair degree $dd_{\max} = rd_{\max} = 5\%$. Link scores are derived from link bus frequencies and re-addition of the same links in the same step is not allowed. The sample size of link set L, that are tested in sub-network evaluations before every addition (Line 7 in Algorithm 5), is set to 10. The sub-networks used are composed of 4-links distance upstream and downstream of the central link (as in Fig. 2)) and traffic simulation is done for a 2-h peak period (from 0.5 to 4.5 h). The initial state of the sub-networks is imported by the most recent full-network simulation. The characteristics of the best found solution set are listed in Table 1. While LNS with sub-networks did not provide significantly better results in this case study and performs worse than other LNS approaches without sub-networks, it can be of added value for even larger networks where a full simulation of the network might be prohibited; for example Chicago city has a network of 64000 links and close to 21000 buses (see Verbas et al., 2015), compared to only 426 links and 522 buses for the case study considered in this paper.

5.4. Best found solutions

By considering the installation/maintenance cost in the objective function, the optimization process identifies the amount of





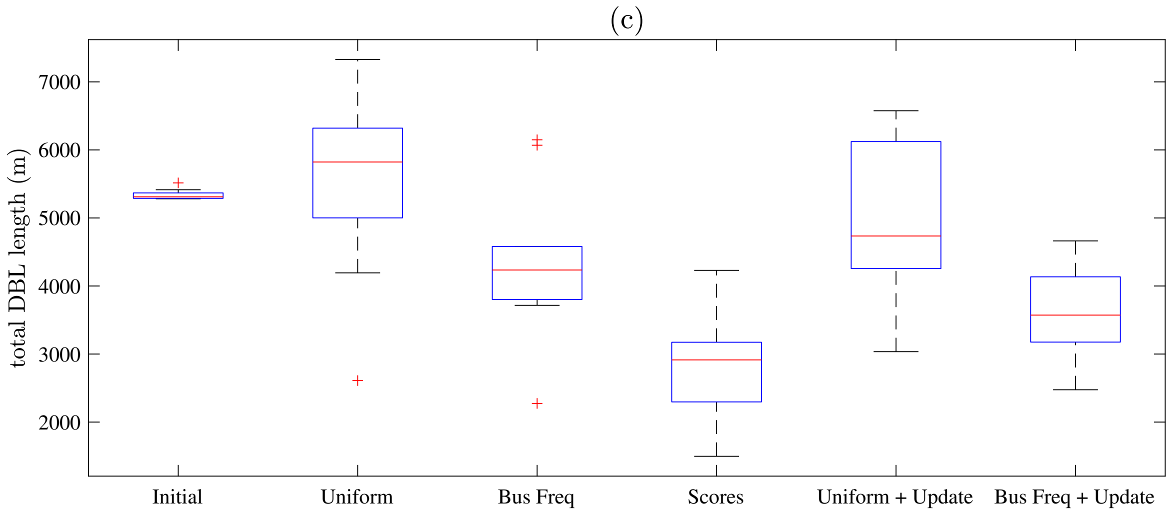


Fig. 9. Performance of LNS algorithms for every type of link scores used. The boxplots show the distribution of different attributes of the best found and the initial solutions for each case; (a) distribution of total cost (PHT + penalty cost); (b) distribution of total travel time (PHT); (c) distribution of total DBL length.

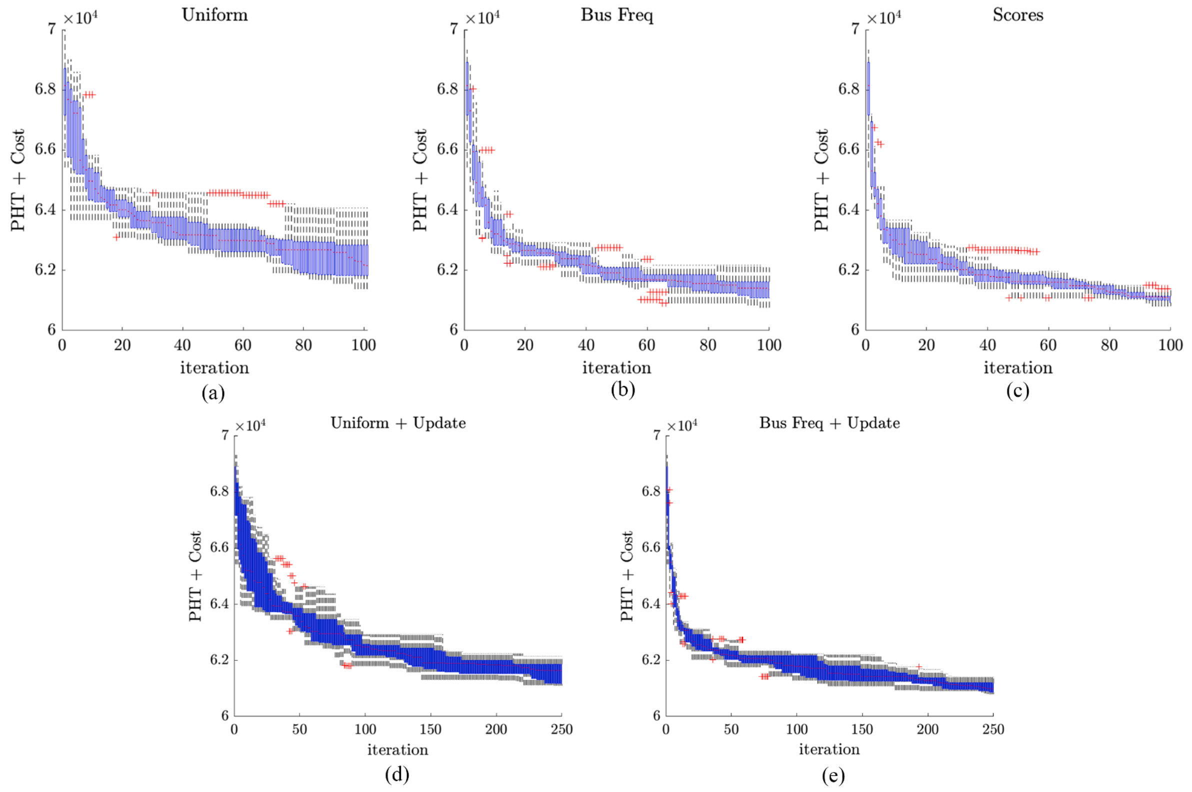


Fig. 10. Evolution of the search process of LNS algorithm with different types of link scores; (a) Uniform; (b) based on bus frequency; (c) based on simulation trials (Alg. 1); (d) Uniform with update process; (e) based on bus frequency with update process. Figures show distribution of total cost of the best solution found so far for all performed replications in the form of boxplots.

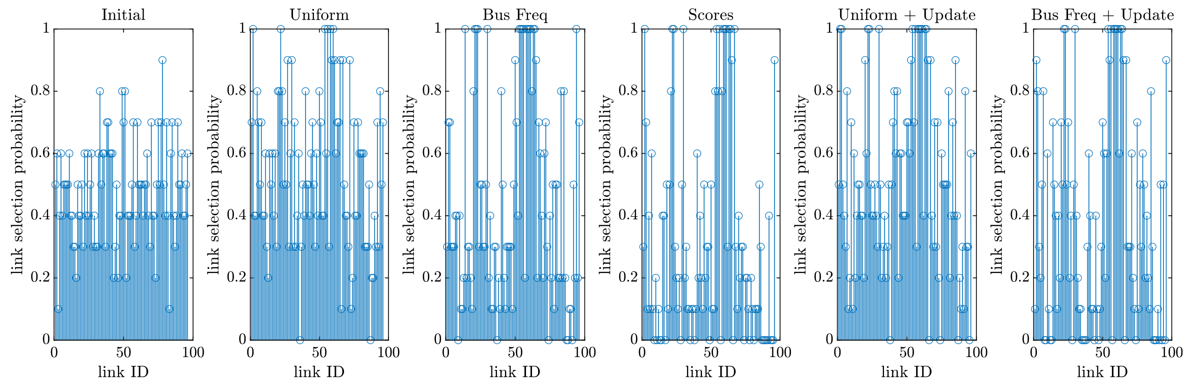


Fig. 11. Selection probability of all candidate links in the Initial and the best found set of solutions for every type of link scores used by LNS algorithm, for all performed replications per case.

required lane-kilometers of DBL indirectly, by balancing delay savings and costs. Setting a suitable value for cost factor γ can be done by accounting several maintenance cost components and the process depends on the specific case study and the objectives of the traffic authorities. In cases where cost is not important, the decision making process can be facilitated by applying LNS algorithm for problem variant 1, i.e. considering $\gamma = 0$ and introducing a constraint for a target total solution length. In fact, by solving the problem for a range of target length values, we can construct a Pareto frontier, as in Fig. 12, demonstrating the improvement in PHT as a function of the sum of lane-kilometers occupied by DBLs.

Adding to the cases tested for variant 2 that were discussed in the previous section, LNS algorithm is applied for problem variant 1 as well, by using link scores defined by Algorithm 1 and target length constraint corresponding to 25% and 50% of the total candidate lane-kilometers, as well as for a case where neither a penalty cost term nor a length constraint is included. The characteristics of the best solution found per case are listed in Table 1. The first six columns of the table list the percent change with respect to the initial

Table 1
Performance of the best solutions found by LNS Algorithm 2 by using different link score types (upper part) and sub-network evaluations (middle part) for problem variant 1, and by using link scores defined by Algorithm 1 for problem variant 2 (lower part). The values represent the % change with respect to the values observed in initial scenario (no DBL), except for the last two columns, which show total DBL length (km) and median execution time (h) per replication per case, respectively.

Case	PHT + Cost	PHT	car	bus	TT_c/km	TT_b/km	DBL (km)	Time (h)
Uniform	−11.36	−16.75	−5.73	+15.77	−11.79	−26.46	5.00	1.57
Bus Freq	−12.50	−16.94	−5.24	+14.42	−12.46	−25.53	4.11	1.59
Scores	−12.52	−14.82	−4.21	+11.59	−11.17	−21.96	2.13	1.34
Uniform + update	−11.90	−15.17	−4.53	+12.48	−11.23	−22.98	3.03	3.45
Bus Freq + update	−12.57	−17.03	−5.57	+15.33	−12.24	−26.31	4.13	2.80
Scores + sub/works	−11.48	−18.27	−6.40	+17.62	−12.78	−28.82	6.30	2.48
PHT + const. 25 %	−12.56	−15.53	−4.34	+11.95	−11.79	−23.03	2.76	0.85
PHT + const. 50 %	−11.84	−17.93	−5.57	+15.34	−13.16	−27.46	5.64	1.04
PHT + no const.	−8.59	−19.45	−7.74	+21.31	−12.79	−32.10	10.07	1.92

network state (no DBL) in: PHT values with and without adding the penalty cost for the total lane-kilometers of DBL (as set in Section 5.2, point 2); car and bus users preferences after DBL introduction; and average travel time per kilometer TT_c and TT_b for car and bus, respectively. The last two columns list the total number of DBL lane-kilometers of each plan and the median execution time per replication per case, respectively. The optimization algorithms, as well as simulation model, are coded in Matlab R2019b and run on an Intel-Core i7-7700 CPU at 3.6 GHz. We observe that all solutions found through the proposed framework lead to a significant PHT improvement, in the range of 15% to 19% with respect to the no DBL case. The solution leading to the highest possible improvement in total PHT is found by LNS algorithm when no length constraint or penalty cost is considered. This solution, however, is the least efficient as the algorithm tends to add a lot of lane-kilometers of DBL, mainly due to the continuing, assumed mode shift from cars to buses that is predicted by the mode choice model. Nevertheless, high accuracy of the prediction of the induced mode shift cannot be guaranteed, even with a well calibrated model, as commuters' behaviour can be highly influenced by numerous factors that might not be considered by the model. This is why, the efficiency of a DBL plan in terms of lane-km requirement should be considered, meaning that DBL plan should aim at a high ratio of PHT improvement over road space usage. For instance, DBL plans found in case 'Scores' or 'Bus frequencies + update' are the most efficient choices for the present case study, as demonstrated by the respective values of PHT combined with penalty cost.

Best found solutions for all test cases are presented in Fig. 12 where one can observe the relation between achieved PHT improvement compared to the no-DBL case, and total solution length. The points of lower surrounding curve form the Pareto frontier for this case study, which can assist traffic authorities in the decision making process of DBL allocation. As expected, higher number of DBL lane-kilometers correspond to higher improvement in total travel time. The most efficient solutions seem to be those closer to total lengths of 2 to 3 km, after which, the required road space for DBL per unit of additional PHT improvement increases significantly. Algorithm 1 is proved very efficient in finding solutions of good quality but with a relatively high computational cost (25.39 h), which grows exponentially with the size of the network. LNS algorithm applied for problem variant 1 shows remarkable performance, as the best solutions found from all tests are very close or even better that those produced by Algorithm 1, with much less computational cost, according to the values in the last column of Table 1. Regarding the observed execution time, the higher values reported in the cases that include a score update process and in the one using sub-networks, relate to the increased number of performed LNS steps (250 and

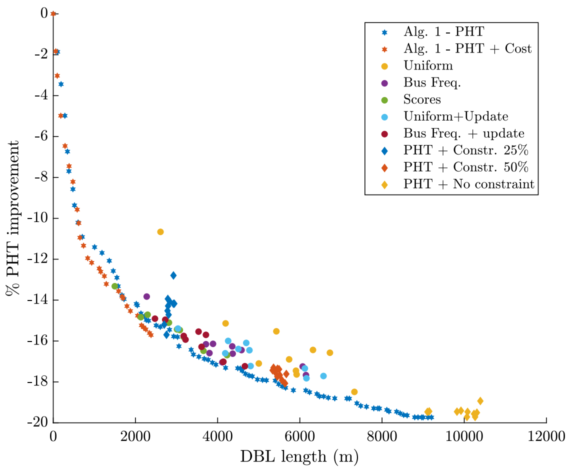


Fig. 12. Pareto frontier relating the % improvement in total PHT compared to the no DBL case, with the DBL length of the best solutions found by Algorithm 1 and LNS Algorithm 2 for the two problem variants.

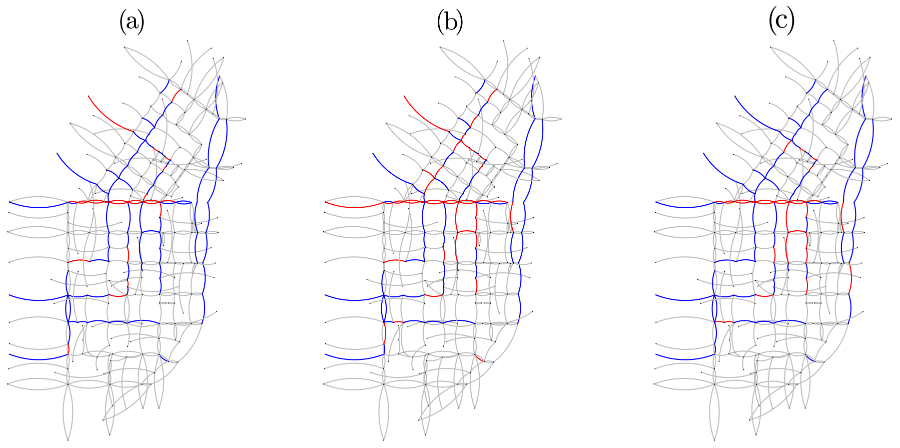


Fig. 13. Spatial distribution of three of the best performing DBL networks identified by different runs of LNS algorithm (links with DBL shown in red) with the respective PHT improvement compared to the initial state of no DBL; (a) LNS using simulation-based scores for problem variant 2 (−12.52%); (b) LNS using scores based on bus frequencies plus an update process for problem variant 2 (−12.57%); (c) LNS using simulation-based link scores for problem variant 1 with a constraint of reserving maximum 25% of available road space (−12.56%).

200, respectively), compared to the rest of the cases, which all performed 100 LNS steps. It is worth highlighting here the difference in computational costs between the proposed method and microscopic simulation. While one complete replication of LNS optimization, performing 100 evaluations of different solutions, lasts on average 1.5-2 h, microscopic simulation might take 30–40 min to evaluate just one candidate solution.

The DBL space distribution of the most promising solutions of Table 1 can be seen in Fig. 13. As expected, links with high bus frequencies are almost always included in the DBL plan. However, this characteristic should not be seen as universal, as the amount of existing car traffic in the same links is a significant contributing factor, which can alter the result in a different case study. It should also be noted that, despite the lack of any type of connectivity constraint, DBLs appear mostly connected in most solutions found. This fact can support the idea that connected solutions are generally more efficient, although this is only implicitly considered by the optimization process, as no connectivity constraint was imposed.

6. Discussion

The paper studies the problem of optimal DBL allocation, in large-scale, bi-modal urban traffic networks, with the aim of minimizing total passenger travel time. We construct a modeling and optimization framework on the basis of a link-based traffic simulation model with queuing characteristics, which is consistent with the dynamic nature of congestion propagation. Mode choice of commuters is adjusted according to travel times that result from the corresponding DBL network configuration, through the use of a simple Logit model. A combinatorial optimization problem is formulated and a set of heuristic and metaheuristic algorithms based on LNS is executed, that can be combined with a learning process and a network decomposition technique, are proposed and tested in terms of effectiveness and efficiency in finding good quality solutions in reasonable time.

Numerical results of application of the proposed scheme in a large network of a real city center show significant potential improvement in the system performance compared to the case of no DBL, proving the effectiveness of the proposed methodology in addressing the DBL allocation problem in large-scale networks considering dynamic congestion. The proposed heuristic for link-by-link construction of DBL plan (Algorithm 1) is shown particularly efficient for the construction of a good quality solution and estimation of link scores, even though its high computational cost might hinder its use in large scale networks. The formulated LNS algorithm is also shown effective in identifying good quality solutions in short time and its efficiency can be increased by proper setting of link selection probabilities (scores) or by including an update process during LNS execution. The proposed update process, similar in concept to A-LNS (see Ropke and Pisinger, 2006), proves efficient even when the initial link selection probabilities are not well tuned, which facilitates the application of LNS. Moreover, the optimized DBL plans are also efficient with respect to the relation between the amount of reserved road space and system performance improvement. Intelligent road space allocation schemes are identified, which can lead to improvement of the system traffic performance even without considering the resulting mode shift from car to bus. This is achieved by identifying candidate DBL locations that improve bus travel time while causing the least possible disturbance to regular traffic, e.g. in wide roads with low traffic flow. A Pareto frontier describing the decrease of the experienced passenger travel time in relation to the road space occupied by DBLs can be created through application of the proposed algorithmic scheme, which can actively support the decision making process regarding DBL network design, depending on the specific objectives and requirements of each case study.

Further research is necessary in order to include the effect of potential re-routing of car users related to DBL introduction. Moreover, combining DBL assignment with other PTP strategies using the same dynamic modeling framework would be an important step towards maximizing transit priority in congested urban networks. Finally, implementation of dynamic bus lanes, that are activated based on congestion evolution and bus presence, as well as combined optimization of DBL distribution and TSP, possibly in parallel with perimeter control schemes, would lead to significant advancement towards reactive intelligent transit systems, that would highly improve mobility especially in highly-congested, large-scale, urban networks.

CRediT authorship contribution statement

Dimitrios Tsitsokas: Conceptualization, Methodology, Investigation, Validation, Writing - review & editing, Software, Writing - original draft. **Anastasios Kouvelas:** Conceptualization, Methodology, Investigation, Validation, Writing - review & editing, Supervision. **Nikolas Geroliminis:** Conceptualization, Methodology, Investigation, Validation, Writing - review & editing, Supervision, Project administration, Funding acquisition.

Acknowledgement

This work was supported by the Swiss National Science Foundation (SNSF) under the project name DIPLOMAT, contract No. 205121-165644.

References

Abdelghany, K.F., Mahmassani, H.S., Abdelghany, A.F., 2007. A modeling framework for bus rapid transit operations evaluation and service planning. Transport. Plann. Technol. 30 (6), 571–591.
Aboudolas, K., Papageorgiou, M., Kosmatopoulos, E., 2009. Store-and-forward based methods for the signal control problem in large-scale congested urban road networks. Transport. Res. Part C: Emerg. Technol. 17 (2), 163–174.
Ampountolas, K., Zheng, N., Geroliminis, N., 2017. Macroscopic modelling and robust control of bi-modal multi-region urban road networks. Transport. Res. Part B: Methodol. 104, 616–637.
Anderson, P., Geroliminis, N., 2020. Dynamic lane restrictions on congested arterials. Transport. Res. Part A: Policy Practice 135, 224–243.
Arasan, V.T., Vedagiri, P., 2010. Microsimulation study of the effect of exclusive bus lanes on heterogeneous traffic flow. J. Urban Plann. Develop. 136 (1), 50–58.
Basso, L.J., Guevara, C.A., Gschwender, A., Fuster, M., 2011. Congestion pricing, transit subsidies and dedicated bus lanes: Efficient and practical solutions to congestion. Transp. Policy 18 (5), 676–684.
Bayrak, M., Guler, S.I., 2018. Optimizing bus lane placement on networks while accounting for queue spillbacks. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC). IEEE, pp. 920–925.
Chen, X., Yu, L., Zhu, L., Guo, J., Sun, M., 2010. Microscopic traffic simulation approach to the capacity impact analysis of weaving sections for the exclusive bus lanes on an urban expressway. J. Transport. Eng. 136 (10), 895–902.
Chiabaut, N., 2015. Evaluation of a multimodal urban arterial: The passenger macroscopic fundamental diagram. Transport. Res. Part B: Methodol. 81, 410–420.
Choi, D., Choi, W., 1995. Effects of an exclusive bus lane for the oversaturated freeway in korea. In: 1995 Compendium of Technical Papers. Institute of Transportation Engineers 65th Annual Meeting. Institute of Transportation Engineers (ITE).
Christofa, E., Ampountolas, K., Skabardonis, A., 2016. Arterial traffic signal optimization: A person-based approach. Transport. Res. Part C: Emerging Technol. 66, 27–47.
Christofa, E., Papamichail, I., Skabardonis, A., 2013. Person-based traffic responsive signal control optimization. IEEE Trans. Intell. Transp. Syst. 14 (3), 1278–1289.
Dahlgren, J., 1998. High occupancy vehicle lanes: Not always more effective than general purpose lanes. Transport. Res. Part A: Policy Practice 32 (2), 99–114.
Deng, T., Nelson, J.D., 2011. Recent developments in bus rapid transit: a review of the literature. Transport Rev. 31 (1), 69–96.
Diakaki, C., Papageorgiou, M., Dinopoulou, V., Papamichail, I., Garyfalia, M., 2014. State-of-the-art and-practice review of public transport priority strategies. IET Intel. Transport Syst. 9 (4), 391–406.
Eichler, M., Daganzo, C.F., 2006. Bus lanes with intermittent priority: Strategy formulae and an evaluation. Transport. Res. Part B: Methodol. 40 (9), 731–744.
Farid, Y.Z., Christofa, E., Collura, J., 2015. Dedicated bus and queue jumper lanes at signalized intersections with nearside bus stops: Person-based evaluation. Transp. Res. Rec. 2484 (1), 182–192.
Farid, Y.Z., Christofa, E., Collura, J., 2018. An analytical model to conduct a person-based evaluation of transit preferential treatments on signalized arterials. Transport. Res. Part C: Emerg. Technol. 90, 411–432.
Geroliminis, N., Zheng, N., Ampountolas, K., 2014. A three-dimensional macroscopic fundamental diagram for mixed bi-modal urban networks. Transport. Res. Part C: Emerg. Technol. 42, 168–181.
Gonzales, E.J., Geroliminis, N., Cassidy, M.J., Daganzo, C.F., 2010. On the allocation of city space to multiple transport modes. Transport. Plann. Technol. 33 (8), 643–656.
Guler, S.I., Cassidy, M.J., 2012. Strategies for sharing bottleneck capacity among buses and cars. Transport. Res. Part B: Methodol. 46 (10), 1334–1345.
Guler, S.I., Gayah, V.V., Menendez, M., 2016. Bus priority at signalized intersections with single-lane approaches: A novel pre-signal strategy. Transport. Res. Part C: Emerg. Technol. 63, 51–70.
Khoo, H.L., Teoh, L.E., Meng, Q., 2014. A bi-objective optimization approach for exclusive bus lane selection and scheduling design. Eng. Optim. 46 (7), 987–1007.
Kouvelas, A., Lioris, J., Fayazi, S., Varaiya, P., 2014. Maximum pressure controller for stabilizing queues in signalized arterial networks. Transport. Rese. Rec.: J. Transport. Res. Board 2421, 133–141.
Levinson, H.S., Zimmerman, S., Clinger, J., Gast, J., 2003. Bus rapid transit: Synthesis of case studies. Transp. Res. Rec. 1841 (1), 1–11.
Li, S., Ju, Y., 2009. Evaluation of bus-exclusive lanes. IEEE Trans. Intell. Transp. Syst. 10 (2), 236–245.
Lin, S., De Schutter, B., Xi, Y., Hellendoorn, H., 2011. Fast model predictive control for urban road networks via milp. IEEE Trans. Intell. Transp. Syst. 12 (3), 846–856.
Lin, S., De Schutter, B., Xi, Y., Hellendoorn, H., 2012. Efficient network-wide model-based predictive control for urban traffic networks. Transport. Res. Part C: Emerg. Technol. 24, 122–140.
Meng, Q., Qu, X., 2013. Bus dwell time estimation at bus bays: A probabilistic approach. Transport. Res. Part C: Emerg. Technol. 36, 61–71.
Mesbah, M., Sarvi, M., Currie, G., 2011. Optimization of transit priority in the transportation network using a genetic algorithm. IEEE Trans. Intell. Transp. Syst. 12 (3), 908–919.
Miandoabchi, E., Farahani, R.Z., Szeto, W.Y., 2012. Bi-objective bimodal urban road network design using hybrid metaheuristics. CEJOR 20 (4), 583–621.
NRC, 2010. Highway capacity manual. Transportation Research Board, The National Academy of Sciences.
Pisinger, D., Ropke, S., 2019. Large neighborhood search. In: Handbook of Metaheuristics. Springer, pp. 99–127.
Ropke, S., Pisinger, D., 2006. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transport. Sci. 40 (4), 455–472.
Shalaby, A.S., 1999. Simulating performance impacts of bus lanes and supporting measures. J. Transport. Eng. 125 (5), 390–397.
Shaw, P., 1998. Using constraint programming and local search methods to solve vehicle routing problems. In: International Conference on Principles and Practice of Constraint Programming. Springer, pp. 417–431.
Stirzaker, C., Dia, H., 2007. Evaluation of transportation infrastructure management strategies using microscopic traffic simulation. J. Infrastruct. Syst. 13 (2), 168–174.

Sun, X., Wu, J., 2017. Combinatorial optimization of bus lane infrastructure layout and bus operation management. Adv. Mech. Eng. 9 (9), 1687814017703341.
Toth, G., 2007. Reducing growth in vehicle miles traveled: Can we really pull it off? In: Driving Climate Change. Elsevier, pp. 129–142.
Verbas, İ.Ö., Mahmassani, H.S., Hyland, M.F., 2015. Dynamic assignment-simulation methodology for multimodal urban transit networks. Transp. Res. Rec. 2498 (1), 64–74.
Viegas, J., Lu, B., 2001. Widening the scope for bus priority with intermittent bus lanes. Transport. Plann. Technol. 24 (2), 87–110.
Waterson, B., Rajbhandari, B., Hounsell, N., 2003. Simulating the impacts of strong bus priority measures. J. Transport. Eng. 129 (6), 642–647.
Wei, L., Chong, T., 2002. Theory and practice of bus lane operation in kunming. DISP-The Plann. Rev. 38 (151), 68–72.
Wirasinghe, S., Kattan, L., Rahman, M., Hubbell, J., Thilakaratne, R., Anowar, S., 2013. Bus rapid transit–a review. Int. J. Urban Sci. 17 (1), 1–31.
Yao, J., Shi, F., Zhou, Z., Qin, J., 2012. Combinatorial optimization of exclusive bus lanes and bus frequencies in multi-modal transportation network. J. Transport. Eng. 138 (12), 1422–1429.
Yu, B., Kong, L., Sun, Y., Yao, B., Gao, Z., 2015. A bi-level programming for bus lane network design. Transport. Res. Part C: Emerg. Technol. 55, 310–327.
Zhang, F., Zheng, N., Yang, H., Geroliminis, N., 2018. A systematic analysis of multimodal transport systems with road space distribution and responsive bus service. Transport. Res. Part C: Emerg. Technol. 96, 208–230.
Zhao, J., Yu, J., Xia, X., Ye, J., Yuan, Y., 2019. Exclusive bus lane network design: A perspective from intersection operational dynamics. Networks Spatial Econ. 19 (4), 1143–1171.
Zheng, N., Geroliminis, N., 2013. On the distribution of urban road space for multimodal congested networks. Transport. Res. Part B: Methodol. 57, 326–341.
Zockaie, A., Saberi, M., Saedi, R., 2018. A resource allocation problem to estimate network fundamental diagram in heterogeneous networks: Optimal locating of fixed measurement points and sampling of probe trajectories. Transport. Res. Part C: Emerging Technol. 86, 245–262.