A spatial branch-and-price-and-cut algorithm for finding globally optimal solutions to the continuous network design problem

Michael W. Levin[a], David Rey[b]

December 9, 2025

[a]Associate Professor, Department of Civil, Environmental, and Geo- Engineering, University of Minnesota, 500 Pillsbury Dr SE, Minneapolis, MN 55455, mlevin@umn.edu

[b]Professor, SKEMA Business School, Universitè Côte D'Azur, Sophia, 60 Rue Fedor Dostoïevski CS 30085, 06902 Antibes Cedex, France, david.rey@skema.edu

Abstract

Transportation network design, or the problem of optimizing infrastructure for a societal goal, subject to individual travelers optimizing their behavior for their own preferences arises frequently in many contexts. However, it is also an NP-hard problem due to the leader-follower or bi-level structure involving a follower objective that is different from yet significantly affects the leader objective. Creating globally optimal solution algorithms has been particularly difficult for the continuous network design problem (CNDP), in which leader variables are continuous, because of the many nonlinearities arising therein. We present a globally optimal algorithm for the CNDP based on using the high-point relaxation, i.e., the system optimal CNDP, and value function cuts to find lower bounds and solving the traffic assignment follower problem to find upper bounds. We introduce a convex relaxation and a spatial branching scheme to handle the non-convexity of value function cuts. This leads to a spatial branch-and-cut algorithm that gradually cuts out bi-level infeasible points from the feasible region of the CNDP. To enhance computational performance, we outer-approximate nonlinear convex functions and use column generation to obtain a sequence of linear programs that can be solved relatively quickly. We show that, for a predefined ϵ, our spatial branch-and-price-and-cut algorithm converges to ϵ-optimality. Compared to prior work on globally optimal solution algorithms for CNDP, we can find ϵ-optimal solutions for the same small test networks in much less time, or solve CNDP on problem instances based on networks that are two orders of magnitude larger than those used in the literature.

Keywords: continuous network design problem; outer approximation; value function cuts; traffic assignment; user equilibrium

1 Introduction

The transportation network design problem (NDP) (Farahani et al., 2013) is the problem of deciding where to improve road network infrastructure so as to optimize some societal goal such as minimizing traffic congestion. The difficulty in solving NDP comes from the fact that the effects of new infrastructure depend on drivers' behavior, and drivers usually behave according to their own preferences such as minimizing their individual travel times. The Braess (1968) paradox illustrates how well-intentioned network design decisions could actually make the network worse off due to user equilibrium route choice. In other words, infrastructure design decisions must anticipate drivers' behavior to achieve a targeted societal goal. This creates a leader-follower game that can be represented as a bi-level optimization problem, which was first identified by Leblanc (1975). Although the classic variant of the problem involves building new road network links or adding capacity to existing links, network design remains highly relevant with emerging technologies. For automated vehicles, researchers have studied which lanes, roads, or regions should be dedicated to automated vehicles (Chen et al., 2017; Levin et al., 2020; Seilabi et al., 2023). For electric vehicles, researchers have studied where to locate charging stations (Chen et al., 2016; Wang et al., 2023; Mirheli and Hajibabai, 2023).

NDPs can be categorized into discrete NDP (DNDP), where the decision variables are discrete, or continuous NDP (CNDP), where the decision variables can take on any value within some continuous range. The different types of decision variables have a significant effect on globally optimal solution algorithms. In DNDP, the discreteness means that the feasible region of the design decision variables is finite, which can be used to guarantee algorithm termination and global optimality. In CNDP, the continuous nature of leader variables together with the non-linear non-convex follower optimality conditions forbids such approaches. Consequently, existing globally optimal solution algorithms in the literature for the CNDP are computationally intractable on even small problem instances. For example, the small Sioux Falls test network has not been solved to global optimality.

Due to the bi-level structure, NDP is non-convex even if the leader objective and follower objective are convex, and NP-hard to solve (Gairing et al., 2017). Therefore, finding globally optimal solutions is computationally difficult, and researchers who encounter NDP often use metaheuristics such as genetic algorithms (Nayeem et al., 2014; Arbex and da Cunha, 2015; Levin et al., 2020). Other recent work has developed methods proven to converge to a stationary point (Wang et al., 2022; Guo et al., 2025). However, although it is possible that the stationary point found by such algorithms is a global optimum, it is not guaranteed and prior work did not attempt to prove it. Our goal is to find a globally optimal solution and prove that it is ϵ away from global optimality.

Recent approaches to finding globally optimal solutions to CNDP cannot scale to large networks. Wang and Lo (2010) formulated the CNDP as a single-level mixed integer linear program (MILP), using binary variables for complementary slackness to enforce the user equilibrium behavior. Because travel time functions are typically nonlinear, they used a piecewise-linear approximation. The large number of integer variables and the necessity of enumerating path flow variables prevents finding solutions on large networks. Luathep et al.

(2011) used a link-based complementary slackness condition to avoid path enumeration, but a MILP on multicommodity link flows still requires substantial computation time for large networks. Du and Wang (2016) used the same complementary slackness formulation, but used geometric programming to handle its nonlinearity. However, that approach introduces difficulties with the linear constraints, and still requires enumerating all paths. Li et al. (2012) does not require path enumeration, but instead solve a sequence of concave minimization problems. Each subproblem is therefore NP-hard. The reported computation times from these recent methods on the 6 node, 16 link network of Harker and Friesz (1984) range from 1.5 minutes using Wang and Lo (2010)'s piecewise approximation to 12 hours from Li et al. (2012). Globally optimal solutions on larger networks have not been reported in the literature.

Since the CNDP is a continuous non-convex optimization problem, it is appropriate to consider the concept of ϵ-optimality which is widely used in global optimization (Liberti, 2008; Floudas, 2013) as a target for CNDP algorithms. The goal of this paper is to develop a new solution algorithm for the CNDP that, given a predefined $\epsilon > 0$, can find ϵ-optimal solutions much more efficiently and on significantly larger problem instances than prior work. These prior methods involve complex subproblems that are difficult to solve, and do not take advantage of the relatively simple subproblems that CNDP is related to. First, for any fixed leader variable, the follower problem of traffic assignment can be solved efficiently on large networks using dedicated algorithms (e.g. Dial, 2006; Bar-Gera, 2010). Second, in many use cases (e.g. the Bureau of Public Roads travel time function), the high-point relaxation of CNDP has a convex objective function and linear constraints (Beckmann et al., 1956). By adding value function cuts to enforce follower optimality, we can form a single-level non-convex nonlinear program that is equivalent to CNDP. We relax the non-convex value function constraint with convex lower and upper bounds, and use spatial branching to improve the tightness of the relaxation to iteratively converge to ϵ-optimality. To enhance computational tractability we also integrate a column generation approach that allows us to exploit the path-based formulation of traffic assignment within our approach.

The contributions of this paper are as follows: we present a new spatial branch-and-price-and-cut algorithm for the CNDP that is guaranteed to converge to ϵ-optimal solutions. Our approach exploits the high-point relaxation of the bi-level formulation and uses spatial branching and value function cuts to gradually tighten this relaxation and obtain increasing lower bounds. Computational tractability is further enhanced using outer approximation (OA) to form a linear approximation of this problem and a column generation procedure to scale up to larger networks. As this algorithm involves solving linear programs and traffic assignment, it is far more efficient than existing methods, which we demonstrate on both the Harker and Friesz (1984) network used in prior CNDP work and on much larger networks.

The remainder of this paper is organized as follows. Section 2 reviews the literature on network design. We formally define the CNDP in Section 3 and present our proposed solution method in Section 4. We show numerical comparisons in Section 5, and conclude in Section 6.

2 Literature review

The discrete network design problem (DNDP) was first proposed by Leblanc (1975), which was followed soon after by the continuous network design problem (CNDP) by Abdulaal and LeBlanc (1979). Although Leblanc (1975) proposed a branch-and-bound algorithm for the DNDP, Abdulaal and LeBlanc (1979) proposed grid-based methods for the CNDP. This theme of difficulty for solving the CNDP exactly has continued throughout the literature on network design. Although CNDP has significant differences from DNDP, some concepts and theory from DNDP are relevant to CNDP and our proposed algorithm. Hence, we first review approaches for the DNDP then review those for the CNDP.

2.1 Discrete network design problem

As is typical of integer programming, DNDP algorithms are often based on finding lower bounds from the high-point relaxation (which corresponds to a system optimal variant of the problem), branching on discrete variables and comparing with upper bounds from solving the traffic assignment follower problem. Leblanc (1975)'s seminal branch-and-bound algorithm for the DNDP adopted this principle. Because Leblanc (1975)'s algorithm relies heavily on traffic assignment to obtain upper and lower bounds, its computation time has improved with the development of faster dedicated algorithms for traffic assignment.

Several studies used piecewise-linear approximations to handle the nonlinear travel time functions, even though the approximation introduces some error. Farvaresh and Sepehri (2011) developed a MILP using a piecewise linear approximation and a linearized version of the user equilibrium condition. Fontaine and Minner (2014) revised the linear approximation to use strong duality to achieve follower optimality. However, instances on Sioux Falls and Berlin Mitte Center networks with 20 discrete leader decision variables remain challenging (Rey, 2020).

Farvaresh and Sepehri (2013) used a similar branch-and-bound approach as Leblanc (1975), but handled the nonlinearity in the high-point relaxation directly using OA (Duran and Grossmann, 1986) instead of branching to obtain fixed leader decision variables. Asadi Bagloee and Sarvi (2018) reformulated the DNDP as a mixed integer nonlinear program (MINLP) to minimize the Beckmann objective function, and used OA there.

Other techniques were also studied for DNDP. Gao et al. (2005) attempted a generalized Benders' decomposition to separate the leader variables from the easier traffic assignment subproblem, but Farvaresh and Sepehri (2013) showed that Gao et al. (2005)'s derivation of the Benders' optimality cuts used the incorrect dual values because both system optimal and user equilibrium traffic assignment are involved. Wang et al. (2013) studied a variation of the original link-addition DNDP of adding discrete levels of link capacity, which created a similar situation as in the standard CNDP in which the follower solution is feasible for all leader decisions. Consequently, their approach exploited value function cuts to gradually enforce follower optimality.

2.2 Continuous network design problem

In contrast to DNDP, designing globally optimal algorithms for CNDP has been more difficult due to the continuous leader variables. Abdulaal and LeBlanc (1979) proposed grid-based methods using a line search or small step-size move. Consequently, after deriving some theoretical results and a single-level reformulation, Marcotte (1986) proposed some heuristics for solving the CNDP. Meng et al. (2001) later proposed a locally-optimal augmented Lagrangian algorithm by reformulating the CNDP using the gap function, which was only guaranteed to find local minima due to the non-convexity of the CNDP. However, one advantage of their algorithm was that its main work involved a variation of traffic assignment, which can be solved quickly. Gao et al. (2007) exploited the similarities of the typical leader and follower objective functions to create a different algorithm, although its global optimality was also not guaranteed. Chiou (2005), Chiou (2007), and Chiou (2009) developed gradient or subgradient methods for the CNDP, but they also have the possibility of becoming stuck in local minima. Ban et al. (2006) created a single-level reformulation using the complementary slackness constraints of the follower traffic assignment problem, which they attempted to solve directly.

Due to the difficulty of solving CNDP to global optimality, several papers have developed methods that are proven to find a local minimum. Wang et al. (2022) proposed a "globally-convergent line search algorithm" which finds a local minimum. Guo et al. (2025) used a difference of convex functions approach to find a stationary point (local minimum), and Guo et al. (2024) used a similar approach to find a stationary point for the related congestion pricing problem. While it is possible that a stationary point found using a convergent algorithm is a global optimal solution, this is not guaranteed, i.e., these globally convergent algorithms are not able to prove global optimality of the obtained solutions. Unlike these studies whose focus was to develop convergent algorithms to find stationary points for the CNDP, our goal is to find a globally optimal solution.

The lack of global optimization approaches for solving the CNDP inspired Wang and Lo (2010) to propose a MILP based on the single-level reformulation obtained using complementary slackness. To achieve linear constraints, the nonlinearity of the travel time function was reformulated with a piecewise-linear approximation. Because their MILP formulation required enumerating all paths, Luathep et al. (2011) extended that approach to mixed continuous-discrete NDP where a link-based complementary slackness condition based on multicommodity link flows is used instead. However, the authors use a piecewise linear approach to handle nonlinear constraints arising in their formulation which does not ensure that a global optimum to the original nonlinear problem is found. Like Luathep et al. (2011), Wang et al. (2015) also reformulated a mixed discrete and continuous network design problem as a single-level MILP, and used outer approximation to create linear versions of nonlinear constraints.

Li et al. (2012) used the gap function to design a globally optimal solution algorithm for the CNDP. Their algorithm requires solving a sequence of concave minimization problems, which are NP-hard, and consequently they reported 12 hours computation times on Harker and Friesz (1984)'s test network, which is commonly used within CNDP studies. Du

and Wang (2016) created another alternative globally optimal approach by reformulating the single-level problem with complementary slackness constraints as a geometric program. However, that required approximating some of the linear constraints relating link and path flows, which created different computational challenges.

Overall, globally optimal methods for the CNDP in the literature are limited, and those that exist require large computation times on small-scale problem instances because they involve solving difficult subproblems. Our method combines elements from the literature on DNDP and CNDP to achieve an algorithm using relatively simple subproblems that admits solutions on much larger networks with reasonable computation times. Specifically, we can solve CNDP to global optimality on much larger networks (over 900 links) in less than 1 hour, whereas prior work has required minutes or hours to solve CNDP to global optimality on the 16 link Harker and Friesz (1984) test network (Li et al., 2012; Du and Wang, 2016).

3 Problem formulation

Although many of the NDPs encountered in specific contexts have context-specific definitions, the CNDP has a standard problem definition from Abdulaal and LeBlanc (1979): deciding how to add capacity to (some) existing links to reduce the combined objective of total system travel time and cost of capacity additions. For the purposes of developing a globally optimal algorithm for the CNDP, it is useful to solve this standard problem, and our algorithm can be modified to apply to other CNDP variants that satisfy certain assumptions discussed hereafter. We will therefore focus on solving the standard CNDP. As in previous work, our main assumptions are based on minimizing total system travel time as the leader objective, with drivers minimizing individual travel time as the follower objective, and the use of mathematical properties satisfied by link congestion function such as the Bureau of Public Roads travel time function.

3.1 Network definition

Consider a traffic network $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ with nodes $\mathcal{N}$ and links $\mathcal{A}$. The set $\mathcal{Z} \subseteq \mathcal{N}$ represents the zones where trips start and end. Some (possibly all) links can be modified with extra capacity. The problem is to decide y_a, the extra capacity added to link $a \in \mathcal{A}$, so as to minimize total congestion. We consider that y_a is lower-bounded by 0 and upper-bounded by Y_a. For links where capacity cannot be added, $Y_a = 0$ and those links are considered as fixed. Each link $a \in \mathcal{A}$ has a travel time function $t_a(x_a, y_a)$ that indicates the travel time on a as a function of link flow x_a and added capacity y_a. For example, the Bureau of Public Roads (BPR) function is

$$t_a(x_a, y_a) = t_a^{\text{ff}} \left(1 + \alpha_a \left(\frac{x_a}{C_a + y_a}\right)^{\rho_a}\right) \tag{1}$$

where C_a is the base capacity, y_a is the added capacity, t_a^{ff} is the free flow travel time, and α_a and ρ_a are calibration constants. Let $\mathbf{x}$ and $\mathbf{y}$ be the vectors of x_a and y_a, respectively.

Assumption 1. *$x_a t_a(x_a, y_a)$ is convex in both x_a and y_a. Moreover, $\int_0^{x_a} t_a(\omega, y_a)d\omega$ is convex in both x_a and y_a.*

Having convex upper-level and lower-level objectives is important for our algorithm design. While Assumption 1 may seem restrictive, Li et al. (2012) proved that it holds for the widely-used BPR travel time function with capacity addition, Eq. (1), and also required this assumption for their globally optimal algorithm. We note that $t_a(x_a, y_a)$ is typically assumed to be increasing in x_a and therefore $\int_0^{x_a} t_a(\omega, y_a)d\omega$ is convex in x_a. Therefore, this assumption is sufficiently general for a wide range of use cases.

Let $g_a(y_a)$ be the cost of adding capacity to link $a \in \mathcal{A}$. This cost function is used in the objective to minimize the weighted combination of total system travel time and costs for added capacity.

Assumption 2. *The functions $g_a(y_a)$, for all $a \in \mathcal{A}$, are convex.*

Under Assumption 2, the objective function of minimizing the combination of congestion and costs for added capacity, is a sum of convex functions and is therefore convex.

$$\min Z(\mathbf{x}, \mathbf{y}) = \sum_{a \in \mathcal{A}} \left(x_a t_a(x_a, y_a) + g_a(y_a)\right) \tag{2}$$

The travel demand from r to s, denoted d_{rs}, is assumed known, and is related to link flows x_a via the flow h^π on path π:

$$x_a = \sum_{\pi \in \Pi} \delta_a^\pi h^\pi \tag{3}$$

where Π is the set of all paths and $\delta_a^\pi \in \{0, 1\}$ indicates whether path π uses link a. All demand must be assigned to a path:

$$\sum_{\pi \in \Pi_{rs}} h^\pi = d_{rs} \tag{4}$$

where $\Pi_{rs} \subseteq \Pi$ is the set of paths from r to s.

Assumption 3. *Drivers behave according to a Wardrop (1952) equilibrium, i.e. link flows satisfy*

$$h^\pi \left[\left(\sum_{a \in \mathcal{A}} \delta_a^\pi t_a(x_a, y_a) \right) - \mu_{rs} \right] = 0 \tag{5}$$

where μ_{rs} is the minimum travel time from r to s. In other words, either $h^\pi = 0$ or the travel time of path π is equal to μ_{rs}.

This creates a leader-follower game where the leader, optimizing link extra capacities $\mathbf{y}$, aims to minimize congestion effects but is subject to the follower behavior of individual drivers which determines link flows $\mathbf{x}$ potentially competing with the leader goals. For example, the Braess (1968) paradox illustrates how a well-intentioned choice of $\mathbf{y}$ could cause $Z(\mathbf{x}, \mathbf{y})$ to increase due to user equilibrium behavior.

3.2 Traffic assignment follower problem

For a fixed $\mathbf{y}$, the follower problem is a $\mathbf{y}$-parameterized user equilibrium traffic assignment problem and can be written as

$$\texttt{TAP}(\mathbf{y}) = \underset{\mathbf{x},\mathbf{h}}{\arg\min} \quad B(\mathbf{x},\mathbf{y}) = \sum_{a\in\mathcal{A}} \int_0^{x_a} t_a(\omega, y_a)d\omega \tag{6a}$$

$$\text{s.t.} \quad x_a = \sum_{\pi\in\Pi} \delta_a^\pi h^\pi \qquad \forall a \in \mathcal{A} \tag{6b}$$

$$\sum_{\pi\in\Pi_{rs}} h^\pi = d_{rs} \qquad \forall (r,s) \in \mathcal{Z}^2 \tag{6c}$$

$$h^\pi \geq 0 \qquad \forall \pi \in \Pi \tag{6d}$$

The objective function $B(\mathbf{x},\mathbf{y})$ is the Beckmann et al. (1956) function, chosen because the KKT conditions match Assumption 3. The specific value of $B(\mathbf{x},\mathbf{y})$ will be useful here in adding cuts during the solution algorithm. From Assumption 1, we assume that $t_a(x_a, y_a)$ is convex in both x_a and y_a, therefore $B(\mathbf{x},\mathbf{y})$ is a sum of convex functions. Therefore the follower problem has convex objective function and linear constraints, and an unique $\mathbf{x}$ solution. While the path flow ($\mathbf{h}$) solution is not unique, only $\mathbf{x}$ is used when calculating the lower-level or upper-level objective functions. We will use $\mathcal{X}$ to denote the feasible space of link flows defined by constraints (6b)–(6d) (which does not depend on $\mathbf{y}$).

In this standard formulation, the choice of $\mathbf{y}$ does not affect the feasibility of $\mathbf{x}$. In other words, y_a decides where to add extra capacity, but a link flow of x_a is feasible even if $y_a = 0$. For standard CNDP this is a typical assumption because y_a indicates the additional capacity to existing links. Problems where the existence of links is determined by the leader variables (e.g. forcing flow to be 0, $x_a = 0$, if $y_a = 0$) require discrete leader decision variables to determine whether the link exists or not. Therefore, for this variation, we refer interested readers to work on solving discrete network design.

3.3 Bi-level formulation of continuous network design problem

The continuous network design problem (CNDP) is

$$\texttt{CNDP}: \min_{\mathbf{x},\mathbf{y}} \quad Z(\mathbf{x},\mathbf{y}) = \sum_{a\in\mathcal{A}} (x_a t_a(x_a, y_a) + g_a(y_a)) \tag{7a}$$

$$\text{s.t.} \quad 0 \leq y_a \leq Y_a \qquad \forall a \in \mathcal{A} \tag{7b}$$

$$\mathbf{x} \in \texttt{TAP}(\mathbf{y}) \tag{7c}$$

which is a bi-level optimization problem. CNDP is primarily difficult due to its bi-level structure. The follower problem itself has been studied extensively, and fast algorithms (e.g. Dial, 2006; Bar-Gera, 2010) exist to solve it on large networks. The bi-level nature makes CNDP a continuous non-convex problem. Our goal is to solve CNDP to ϵ-optimality for a predefined

$\epsilon > 0$. The concept of ϵ-optimality is widely used in global optimization methods for non-convex problems, notably if decision variables are continuous (Liberti, 2008; Floudas, 2013). Formally, we aim to find $(\mathbf{x}^\star, \mathbf{y}^\star)$ such that $Z(\mathbf{x}^\star, \mathbf{y}^\star)$ is within ϵ of the optimal objective function value (OFV) of `CNDP` and $B(\mathbf{x}^\star, \mathbf{y}^\star)$ is within ϵ of $\min_{\mathbf{x}} B(\mathbf{x}, \mathbf{y}^\star)$.

We will start by working with the high-point relaxation of `CNDP`, i.e., the system optimal CNDP (SO-CNDP):

$$\texttt{SO-CNDP}: Z_{\text{lb}} = \min_{\mathbf{x},\mathbf{y},\mathbf{h}} \quad Z(\mathbf{x}, \mathbf{y}) = \sum_{a \in \mathcal{A}} \left(x_a t_a(x_a, y_a) + g_a(y_a)\right) \tag{8a}$$

$$\text{s.t.} \quad 0 \leq y_a \leq Y_a \qquad \forall a \in \mathcal{A} \tag{8b}$$

$$x_a = \sum_{\pi \in \Pi} \delta_a^\pi h^\pi \qquad \forall a \in \mathcal{A} \tag{8c}$$

$$\sum_{\pi \in \Pi_{rs}} h^\pi = d_{rs} \qquad \forall (r,s) \in \mathcal{Z}^2 \tag{8d}$$

$$h^\pi \geq 0 \qquad \forall \pi \in \Pi \tag{8e}$$

Formulation (8) has a convex objective function and linear constraints and can be solved efficiently. However, its solutions are unlikely to be bi-level feasible, i.e., they may not satisfy constraint (7c). Our approach starts by solving `SO-CNDP` and iteratively enforces bi-level feasibility via branching and cutting schemes.

4 Solution algorithm

Our algorithm focuses on computing a sequence of progressively tighter upper and lower bounds, as in several globally optimal algorithms for DNDP (Leblanc, 1975; Farvaresh and Sepehri, 2013). Upper bounds are computed from solving traffic assignment for a fixed leader solution $\mathbf{y}$, whereas lower bounds are found by solving the high-point relaxation problem augmented with value function cuts such as in Wang et al. (2013) to move towards follower optimality.

The structure of our algorithm is as follows: we start by solving `SO-CNDP` and obtaining a global lower bound Z_{lb} of the optimal OFV $Z^\star$ along with a point $(\mathbf{x}^{\text{l}}, \mathbf{y}^{\text{l}})$. An upper bound is obtained by solving $\mathbf{x}^{\text{f}} = \texttt{TAP}(\mathbf{y}^{\text{l}})$ and computing $Z(\mathbf{x}^{\text{f}}, \mathbf{y}^{\text{l}})$. If

$$Z(\texttt{TAP}(\mathbf{y}^{\text{l}}), \mathbf{y}^{\text{l}}) - Z_{\text{lb}} \leq \epsilon \tag{9}$$

then we have found a solution that is ϵ-optimal and we can terminate the search. Otherwise, the algorithm explores the feasible region of `SO-CNDP` by performing spatial branching on $\mathbf{y}$-variables and adding value function cuts based on the follower best-response solution to tighten the relaxation and eliminate bi-level infeasible points. We next present the main components of our approach.

4.1 Value function cuts

Suppose solving the high-point relaxation (8) yields a solution $(\mathbf{x}^{\mathrm{l}}, \mathbf{y}^{\mathrm{l}})$ where $\mathbf{x}^{\mathrm{l}} \neq \texttt{TAP}(\mathbf{y}^{\mathrm{l}})$, i.e., it is bi-level infeasible. Because the follower problem (6) seeks to minimize the Beckmann function $B(\mathbf{x}, \mathbf{y})$, by definition it holds that

$$B\left(\mathbf{x}^{\mathrm{l}}, \mathbf{y}^{\mathrm{l}}\right) > \min_{\mathbf{x} \in \mathcal{X}} B\left(\mathbf{x}, \mathbf{y}^{\mathrm{l}}\right) \tag{10}$$

We add value function cuts to Formulation (8) requiring the opposite of Eq. (10) to restrict $\mathbf{x}$ to follower-optimal points, resulting in the augmented high-point relaxation of `SO-CNDP`:

$$\min_{\mathbf{x},\mathbf{y},\mathbf{h}} \quad Z(\mathbf{x}, \mathbf{y}) = \sum_{a \in \mathcal{A}} \left(x_a t_a(x_a, y_a) + g_a(y_a)\right) \tag{11a}$$

$$\text{s.t.} \quad 0 \leq y_a \leq Y_a \qquad \forall a \in \mathcal{A} \tag{11b}$$

$$x_a = \sum_{\pi \in \Pi} \delta_a^\pi h^\pi \qquad \forall a \in \mathcal{A} \tag{11c}$$

$$\sum_{\pi \in \Pi_{rs}} h^\pi = d_{rs} \qquad \forall (r, s) \in \mathcal{Z}^2 \tag{11d}$$

$$h^\pi \geq 0 \qquad \forall \pi \in \Pi \tag{11e}$$

$$B(\mathbf{x}, \mathbf{y}) \leq B(\mathbf{x}', \mathbf{y}) \qquad \forall \mathbf{x}' \in \mathcal{X} \tag{11f}$$

where constraints (11f) are the value function cuts. Formulation (11) is equivalent to `CNDP` because it rewrites constraint (7c) requiring follower optimality as a comparison of the follower objective function against all feasible $\mathbf{x}'$ in constraint (11f).

4.1.1 Generating value function cuts

It is impractical to consider all $\mathbf{x}' \in \mathcal{X}$ for constraint (11f). Our approach consists of relaxing constraint (11f) and iteratively generating such cuts at points violating this constraint. Let $\mathcal{V} = \{1, \ldots, n^{\mathrm{vf}}\}$ be the index set of $\mathbf{x}^{\mathrm{f}}$ points used in value function cuts, labeled $\mathbf{x}^{\mathrm{f}}(i)$ for $i \in \mathcal{V}$. The set $\mathcal{V}$ will be built sequentially over iterations. In other words, $\mathcal{V}$ provides a method to access all saved points used for value function cuts. After combining Formulation (8) with an index set of value function cuts $\mathcal{V}$, we obtain a partially augmented SO-CNDP formulation

named R-CNDP($\mathscr{V}$):

$$\texttt{R-CNDP}(\mathscr{V}): Z_{\text{lb}}(\mathscr{V}) = \min_{\mathbf{x},\mathbf{y},\mathbf{h}} \quad Z(\mathbf{x},\mathbf{y}) = \sum_{a\in\mathcal{A}} \left(x_a t_a(x_a,y_a) + g_a(y_a)\right) \tag{12a}$$

$$\text{s.t.} \quad 0 \le y_a \le Y_a \qquad \forall a \in \mathcal{A} \tag{12b}$$

$$x_a = \sum_{\pi\in\Pi} \delta_a^\pi h^\pi \qquad \forall a \in \mathcal{A} \tag{12c}$$

$$\sum_{\pi\in\Pi_{rs}} h^\pi = d_{rs} \qquad \forall (r,s) \in \mathcal{Z}^2 \tag{12d}$$

$$h^\pi \ge 0 \qquad \forall \pi \in \Pi \tag{12e}$$

$$B(\mathbf{x},\mathbf{y}) \le B\left(\mathbf{x}^{\text{f}}(i),\mathbf{y}\right) \qquad \forall i \in \mathscr{V} \tag{12f}$$

Note that constraint (12f) has a convex LHS and a convex RHS since $B(\mathbf{x},\mathbf{y})$ is convex in both $\mathbf{x}$ and $\mathbf{y}$. Therefore constraint (12f) is non-convex due to its convex RHS.

Since Formulation (12) contains a subset of the constraints of Formulation (11) which is itself equivalent to CNDP, we have shown the following result.

Proposition 1. *Given an index set of value function cuts $\mathscr{V}$, the optimal OFV of Formulation (12) is a lower-bound on the optimal OFV of CNDP, i.e., $Z_{\text{lb}}(\mathscr{V}) \le Z^\star$.*

The purpose of the value function cuts is to sequentially move from a feasible solution of SO-CNDP to a solution of CNDP. To achieve convergence, adding a value function cut of the form (12f) must cut out bi-level infeasible $(\mathbf{x},\mathbf{y})$-points from the feasible region of SO-CNDP. This is demonstrated in the following result proving that we remove the prior $(\mathbf{x}^{\text{l}},\mathbf{y}^{\text{l}})$ point.

Remark 1. *Let $(\mathbf{x}^{\text{l}},\mathbf{y}^{\text{l}})$ be the solution of R-CNDP($\mathscr{V}$). Let $\mathbf{x}^{\text{f}}(i') = $ TAP$(\mathbf{y}^{\text{l}})$. If $\mathbf{x}^{\text{l}} \notin \arg\min_{\mathbf{x}} B(\mathbf{x},\mathbf{y}^{\text{l}})$, i.e., $\mathbf{x}^{\text{l}}$ is not bi-level feasible, then extending the index set $\mathscr{V}' = \mathscr{V} \cup \{i'\}$ and adding the value function cut:*

$$B(\mathbf{x},\mathbf{y}) \le B\left(\mathbf{x}^{\text{f}}(i'),\mathbf{y}\right) \tag{13}$$

excludes $(\mathbf{x}^{\text{l}},\mathbf{y}^{\text{l}})$ from the feasible region of R-CNDP($\mathscr{V}'$).

Proof. Let $\mathbf{x}^{\text{f}}(i') \in \arg\min_{\mathbf{x}} B(\mathbf{x},\mathbf{y}^{\text{l}})$ (which is unique due to convexity of TAP) and $\mathbf{x}^{\text{l}} \ne \mathbf{x}^{\text{f}}$. Then $B(\mathbf{x}^{\text{l}},\mathbf{y}^{\text{l}}) > B(\mathbf{x}^{\text{f}}(i'),\mathbf{y}^{\text{l}})$ and adding the cut $B(\mathbf{x},\mathbf{y}) \le B(\mathbf{x}^{\text{f}}(i'),\mathbf{y})$ to R-CNDP($\mathscr{V}$) removes $(\mathbf{x}^{\text{l}},\mathbf{y}^{\text{l}})$ from the feasible region of problem R-CNDP($\mathscr{V}'$). □

We note that extending the set $\mathscr{V}$ of value function cuts after solving R-CNDP($\mathscr{V}$) may exclude more points than just the previously obtained solution. Since the newly generated value function cut $B(\mathbf{x},\mathbf{y}) \le B\left(\mathbf{x}^{\text{f}}(i'),\mathbf{y}\right)$ is valid for any $\mathbf{y}$, the feasible region of R-CNDP($\mathscr{V}'$) is reduced to the set of $(\mathbf{x},\mathbf{y})$ points that satisfy this constraint.

Moreover, if we consider some fixed $\mathbf{y}' \ne \mathbf{y}$ and the best corresponding $\mathbf{x}'$ from Formulation (8) defined as

$$\mathbf{x}' \in \underset{\mathbf{x}\in\mathcal{X}: B(\mathbf{x},\mathbf{y}') \le B\left(\mathbf{x}^{\text{f}},\mathbf{y}\right)}{\arg\min} Z(\mathbf{x},\mathbf{y}') \tag{14}$$

it is possible that $Z(\mathbf{x}', \mathbf{y}')$ is relatively small. In this situation, cut (13) directly removes some values of $\mathbf{x}$ from feasibility, but also causes some values of $\mathbf{y}$ to correctly be suboptimal for problem `R-CNDP(`$\mathscr{V}'$`)`.

Sequentially extending the index set of value function cuts $\mathscr{V}$ used within `R-CNDP(`$\mathscr{V}$`)` will eventually cut away the values of $\mathcal{X}$ that are feasible for Formulation (8) but infeasible for `CNDP`. Furthermore, we always obtain a lower bound on `CNDP` by Proposition 1, hence the optimal OFV $Z_{\text{lb}}(\mathscr{V})$ is non-decreasing as set $\mathscr{V}$ is extended, forming a non-decreasing sequence of lower bounds.

Value function cuts of the form (12f) are non-convex due to their right-hand side (RHS) term. These cuts can be viewed as difference of convex functions since by Assumption 1 $B(\mathbf{x}, \mathbf{y})$ is convex in both $\mathbf{x}$ and $\mathbf{y}$ and therefore $B(\mathbf{x}^{\text{f}}(i), \mathbf{y})$ for any index $i \in \mathscr{V}$ is convex in $\mathbf{y}$. As a consequence, Formulation `R-CNDP(`$\mathscr{V}$`)` is non-convex and cannot be solved efficiently by existing optimization solvers. To address this non-convexity, we propose a linear relaxation of the RHS of constraint (12f) and a spatial branching scheme to iteratively refine this linear relaxation.

4.1.2 Relaxation and spatial branching scheme

To design the linear relaxation of the RHS of constraint (12f), we introduce variables $\theta_a \in [0, 1]$ for all links $a \in \mathcal{A}$. Note that for links where no capacity can be added (i.e. $y_a = 0$ is the only choice), we do not need a θ_a variable. Given lower bounds $\underline{\mathbf{y}}$ and upper bounds $\overline{\mathbf{y}}$ satisfying $\underline{\mathbf{y}} \leq \mathbf{y} \leq \overline{\mathbf{y}}$, we can rewrite the RHS of constraint (12f) using a convex combination as

$$
\begin{aligned}
B\left(\mathbf{x}^{\text{f}}(i), \mathbf{y}\right) &= \sum_{a \in \mathcal{A}} \int_0^{x_a^{\text{f}}(i)} t_a(\omega, y_a) d\omega \\
&\leq \sum_{a \in \mathcal{A}} \left[\theta_a \int_0^{x_a^{\text{f}}(i)} t_a(\omega, \overline{y}_a) d\omega + (1 - \theta_a) \int_0^{x_a^{\text{f}}(i)} t_a(\omega, \underline{y}_a) d\omega \right] \qquad \forall i \in \mathscr{V} \qquad \text{(15a)} \\
y_a &= \theta_a \overline{y}_a + (1 - \theta_a) \underline{y}_a \qquad \forall a \in \mathcal{A} \qquad \text{(15b)}
\end{aligned}
$$

In terms of implementation, we can assume that $\underline{y}_a = 0$ and $\overline{y}_a = Y_a$ (an upper bound on added capacity) exist, such as due to limited physical space. Eq. (15a) may not be very tight unless the difference $\overline{y}_a - \underline{y}_a$ becomes small. Therefore, we introduce a spatial branching scheme to iteratively reduce these intervals. Given an interval $\underline{y}_a \leq y_a \leq \overline{y}_a$, we can reduce that difference by splitting that interval into two: $\underline{y}_a \leq y_a \leq \tilde{y}_a$ or $\tilde{y}_a \leq y_a \leq \overline{y}_a$. Each of the smaller intervals $[\underline{y}_a, \tilde{y}_a]$ and $[\tilde{y}_a, \overline{y}_a]$ have a tighter linear approximation to the convex RHS of constraint (12f).

We choose to branch on the link $a \in \mathcal{A}$ such that the gap

$$\Delta_a(\theta_a) = \left[\theta_a \int_0^{x_a^{\mathrm{f}}(i)} t_a(\omega, \overline{y}_a)d\omega + (1-\theta_a)\int_0^{x_a^{\mathrm{f}}(i)} t_a(\omega, \underline{y}_a)d\omega\right] - \int_0^{x_a^{\mathrm{f}}(i)} t_a(\omega, \theta_a\overline{y}_a + (1-\theta_a)\underline{y}_a)d\omega \tag{16}$$

is largest for a given $x_a^{\mathrm{f}}(i)$. Because the left-hand side (LHS) of constraint (15a) is convex, $\Delta_a(\theta_a)$ in Eq. (16) is concave, and the point where $\frac{d\Delta_a(\theta_a)}{d\theta_a} = 0$ is the maximum gap. Branching on that point will minimize the gap of the two new intervals. For the commonly-used BPR travel time function, after choosing link a to branch on, the best $\theta_a^\star$ for a specific $x_a^{\mathrm{f}}(i)$ point is

$$\theta_a^\star(i) = \frac{\xi_2 - C_a - \underline{y}_a}{\overline{y}_a - \underline{y}_a} \tag{17}$$

where ξ_2 is defined as

$$\xi_2 = \sqrt[\rho+1]{\frac{x_a^{\rho+1}}{\xi_1(\rho+1)}} \tag{18}$$

with ξ_1 defined as

$$\xi_1 = \frac{\int_0^{x_a^{\mathrm{f}}(i)} t_a(\omega, \overline{y}_a)d\omega - \int_0^{x_a^{\mathrm{f}}(i)} t_a(\omega, \underline{y}_a)d\omega}{\left(\overline{y}_a - \underline{y}_a\right) t_a^{\mathrm{ff}}\alpha_a(-\rho_a - 1)} \tag{19}$$

With different saved $\mathbf{x}^{\mathrm{f}}(i)$ points used to get closer to bi-level feasibility, we choose the branching link as

$$a = \underset{a\in\mathcal{A},\ i\in\{1,\ldots,n^{\mathrm{vf}}\}}{\arg\max} \Delta_a\left(\theta_a^\star(i)\right) \tag{20}$$

Fortunately, there are two advantages with this branching scheme. First, the augmented high-point relaxation to be solved at each branch node is a linear program which can be solved by commercial solvers. Second, in many DNDP or CNDP variants in the literature, $y_a = 0$ for most links. In other words, the number of links with a leader decision variable that requires branching is relatively small. That is a direct advantage in reducing the number of dimensions of the branching, but it also helps with the value function cuts. Most of the RHS value of constraint (15a) comes from links where $y_a = 0$ and the saved point $x_a^{\mathrm{f}}(i)$ provides a globally optimal value for the Beckmann function. Consequently, a large interval $[\underline{y}_a, \overline{y}_a]$ may still have a numerically small gap for constraint (16), reducing the number of branches required for convergence to a desired optimality gap.

Incorporating this linear relaxation within Formulation `R-CNDP(`$\mathscr{V}$`)`, leads to a program named `LR-CNDP(`$\mathscr{V}$`)`:

$$\texttt{LR-CNDP}(\mathscr{V}): \min_{\mathbf{x},\mathbf{y},\mathbf{h},\boldsymbol{\theta}} \quad Z(\mathbf{x},\mathbf{y}) = \sum_{a\in\mathcal{A}} (x_a t_a(x_a, y_a) + g_a(y_a)) \tag{21a}$$

$$\text{s.t.} \quad x_a = \sum_{\pi\in\Pi} \delta_a^\pi h^\pi \qquad \forall a \in \mathcal{A} \tag{21b}$$

$$\sum_{\pi\in\Pi_{rs}} h^\pi = d_{rs} \qquad \forall (r,s) \in \mathcal{Z}^2 \tag{21c}$$

$$h^\pi \geq 0 \qquad \forall \pi \in \Pi \tag{21d}$$

$$\sum_{a\in\mathcal{A}} \int_0^{x_a} t_a(\omega, y_a) d\omega \leq \sum_{a\in\mathcal{A}} \left[\theta_a \int_0^{x_a^{\mathrm{f}}(i)} t_a(\omega, \overline{y}_a) d\omega + (1-\theta_a) \int_0^{x_a^{\mathrm{f}}(i)} t_a(\omega, \underline{y}_a) d\omega \right] \qquad \forall i \in \mathscr{V} \tag{21e}$$

$$y_a = \theta_a \overline{y}_a + (1-\theta_a)\underline{y}_a \qquad \forall a \in \mathcal{A} \tag{21f}$$

$$0 \leq \theta_a \leq 1 \qquad \forall a \in \mathcal{A} \tag{21g}$$

Our approach consists of solving LR-CNDP($\mathscr{V}$) using the spatial branching scheme described in Eqs. (16)–(20). Solving LR-CNDP($\mathscr{V}$) at the root node of the search tree yields a lower bound on the optimal OFV of CNDP and branching on y_a variables will progressively tighten this lower bound. We note that under Assumption 1, both the objective function of Formulation LR-CNDP($\mathscr{V}$) and the LHS of constraint (21e) are convex. Thus, in this case, Formulation LR-CNDP($\mathscr{V}$) is a convex program that could in principle be solved using classical techniques such as interior-point methods which are implemented in convex optimization solvers (e.g. IPOPT). However, Formulation LR-CNDP($\mathscr{V}$) is path-based and it is impractical to enumerate all paths of a network in general. A possibility would be to reformulate LR-CNDP($\mathscr{V}$) using multicommodity link-based variables instead of path-based variables. However, in the context of transportation network design, such link-based approaches have been shown to be computationally less efficient compared to path-based approaches (Rey and Levin, 2024). Since our goal is to solve CNDP at scale, including problem instances based on large networks, we opt to retain the path-based network model and we propose to linearize the objective function and the LHS of constraint (21e) using OA. This will allow us to obtain a linear relaxation of LR-CNDP($\mathscr{V}$) that is amenable to column generation. We next describe how these OA cuts are derived and generated within a spatial branch-and-price-and-cut algorithm.

4.2 OA of the objective function

The objective function (12a) is nonlinear and convex. We propose to use OA to obtain a more tractable formulation. This is motivated by observing that supporting hyperplanes are always linear underestimators of a convex function (Duran and Grossmann, 1986). Therefore,

by gradually adding supporting hyperplane cuts, we can approximate a nonlinear convex function with linear constraints.

The standard OA of $Z(\mathbf{x}, \mathbf{y})$ involves taking a point $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ and adding the linear cut

$$Z(\hat{\mathbf{x}}, \hat{\mathbf{y}}) + \nabla_{\mathbf{x}} Z(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \cdot (\mathbf{x} - \hat{\mathbf{x}}) + \nabla_{\mathbf{y}} Z(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \cdot (\mathbf{y} - \hat{\mathbf{y}}) \leq Z(\mathbf{x}, \mathbf{y}) \tag{22}$$

The downside of OA is that we need a potentially large number of supporting hyperplanes to form a tight approximation of $Z(\mathbf{x}, \mathbf{y})$. Furthermore, since the $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ points used in the approximation are usually based on prior solutions of Formulation (12), that could involve a large number of iterations, with a large number of linear programs to be solved, before the approximation is tight. To address this challenge, we exploit the link-separability of leader and follower objective functions to develop tighter OA models.

The supporting hyperplane on the LHS of Eq. (22) is centered on $Z(\hat{\mathbf{x}}, \hat{\mathbf{y}})$,

$$Z(\hat{\mathbf{x}}, \hat{\mathbf{y}}) = \sum_{a \in \mathcal{A}} \left(\hat{x}_a t_a(\hat{x}_a, \hat{y}_a) + g_a(\hat{y}_a) \right) \tag{23}$$

and the gradient terms are

$$\nabla Z_{\mathbf{x}}(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \cdot (\mathbf{x} - \hat{\mathbf{x}}, \mathbf{y} - \hat{\mathbf{y}}) = \sum_{a \in \mathcal{A}} (x_a - \hat{x}_a) \left[\hat{x}_a \frac{dt(\hat{x}_a, \hat{y}_a)}{dx_a} + t_a(\hat{x}_a, \hat{y}_a) \right] \tag{24}$$

and

$$\nabla_{\mathbf{y}} Z(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \cdot (\mathbf{y} - \hat{\mathbf{y}}) = \sum_{a \in \mathcal{A}} (y_a - \hat{y}_a) \left[\hat{x}_a \frac{dt(\hat{x}_a, \hat{y}_a)}{dy_a} + \frac{dg(\hat{y}_a)}{dy_a} \right] \tag{25}$$

Observe that both $Z(\mathbf{x}, \mathbf{y})$ and its OA are separable by link. Therefore, instead of using the entire $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ for the OA, we can instead derive a link-based OA for the individual objective function components for each link a. Let ζ_a be the variable used for the OA of the term corresponding to link $a \in \mathcal{A}$. Replacing $Z(\mathbf{x}, \mathbf{y})$ with approximation variable ζ_a in Eq. (22), we obtain

$$Z(\hat{\mathbf{x}}, \hat{\mathbf{y}}) + \nabla_{\mathbf{x}} Z(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \cdot (\mathbf{x} - \hat{\mathbf{x}}) + \nabla_{\mathbf{y}} Z(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \cdot (\mathbf{y} - \hat{\mathbf{y}}) \leq \sum_{a \in \mathcal{A}} \zeta_a \tag{26}$$

as the OA cut with the objective of minimizing $\sum_{a \in \mathcal{A}} \zeta_a$. This cut can be decomposed into link-based cuts of the form:

$$\begin{aligned} \zeta_a \geq \hat{x}_a t_a(\hat{x}_a, \hat{y}_a) + g_a(\hat{y}_a) + (x_a - \hat{x}_a) \left[\hat{x}_a \frac{dt(\hat{x}_a, \hat{y}_a)}{dx_a} + t_a(\hat{x}_a, \hat{y}_a) \right] \\ + (y_a - \hat{y}_a) \left[\hat{x}_a \frac{dt(\hat{x}_a, \hat{y}_a)}{dy_a} + \frac{dg(\hat{y}_a)}{dy_a} \right] \end{aligned} \tag{27}$$

4.3 OA of the LHS of value function cuts

We consider a link-based OA scheme for the LHS of constraint (21e). Let β_a be the variable used for the OA of the term corresponding to link $a \in \mathcal{A}$. We replace the LHS of constraint (21e) with $\sum_{a\in\mathcal{A}} \beta_a$. Given a point $(\mathbf{x}', \mathbf{y}')$ at which OA is performed, we add the constraint

$$\beta_a \geq \int_0^{x'_a} t_a(\omega, y'_a)d\omega + (x_a - x'_a) \cdot t_a(x'_a, y'_a) + (y_a - y'_a) \cdot \int_0^{x'_a} \frac{dt_a(\omega, y_a)}{y_a} d\omega \tag{28}$$

for every link a.

In effect, for every $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ point used to generate an OA cut for either the objective function $Z(\mathbf{x}, \mathbf{y})$ or the LHS of constraint (21e), we instead obtain the equivalent of $|\mathcal{A}|$ cuts on link-based variables ζ_a and β_a. If we have n points for generating OA cuts, $(\hat{\mathbf{x}}(i), \hat{\mathbf{y}}(i))$ for $i = 1 \dots n$, this yields $|\mathcal{A}|^n$ OA cuts since every combination of points, e.g. $(\hat{x}_1(1), \hat{x}_2(2), \dots)$ yields a cut on ζ_a.

4.4 Linear relaxation of LR-CNDP($\mathscr{V}$)

Combining the OA machinery of the objective function of LR-CNDP($\mathscr{V}$) and of the LHS of constraint (21e), we obtain a linear relaxation of LR-CNDP($\mathscr{V}$). Let $\mathscr{C} = \{1, \dots, n^{\text{oa}}\}$ be index set of $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ points used to generate OA cuts on both $Z(\mathbf{x}, \mathbf{y})$ and the LHS of constraint (21e), labeled as $(\hat{\mathbf{x}}(i), \hat{\mathbf{y}}(i))$ for $i = 1 \dots n^{\text{oa}}$. Analogously to value function cuts, OA cuts will be added sequentially over iterations. Then, for given index sets $\mathscr{V}$ of value function cuts and $\mathscr{C}$ of OA cuts, Formulation (29) is a linear relaxation of LR-CNDP($\mathscr{V}$):

$$\min_{\mathbf{y},\mathbf{x},\mathbf{h},\theta,\zeta,\beta} \quad \sum_{a\in\mathcal{A}} \zeta_a \tag{29a}$$

$$\text{s.t.} \quad \zeta_a \geq \hat{x}_a(i) t_a\left(\hat{x}_a(i), \hat{y}_a(i)\right) + g_a\left(\hat{y}_a(i)\right) + \left(x_a - \hat{x}_a(i)\right)\left[\hat{x}_a(i)\frac{dt\left(\hat{x}_a(i), \hat{y}_a(i)\right)}{dx_a} + t_a\left(\hat{x}_a(i), \hat{y}_a(i)\right)\right] + \left(y_a - \hat{y}_a(i)\right)\left[\hat{x}_a(i)\frac{dt\left(\hat{x}_a(i), \hat{y}_a(i)\right)}{dy_a} + \frac{dg\left(\hat{y}_a(i)\right)}{dy_a}\right] \quad \forall a \in \mathcal{A}, \forall i \in \mathscr{C} \tag{29b}$$

$$x_a = \sum_{\pi\in\Pi} \delta_a^\pi h^\pi \quad \forall a \in \mathcal{A} \tag{29c}$$

$$\sum_{\pi\in\Pi_{rs}} h^\pi = d_{rs} \quad \forall (r,s) \in \mathscr{Z}^2 \tag{29d}$$

$$h^\pi \geq 0 \quad \forall \pi \in \Pi \tag{29e}$$

$$\beta_a \geq \int_0^{x_a^{\mathrm{l}}(i)} t_a(\omega, y_a^{\mathrm{l}}(i))d\omega + \left(x_a - x_a^{\mathrm{l}}(i)\right)\cdot t_a(x_a^{\mathrm{l}}(i), y_a^{\mathrm{l}}(i)) + \left(y_a - y_a^{\mathrm{l}}(i)\right)\cdot \int_0^{x_a^{\mathrm{l}}(i)} \frac{dt_a(\omega, y_a)}{y_a} d\omega \quad \forall a \in \mathcal{A}, \forall i \in \mathscr{C} \tag{29f}$$

$$\sum_{a\in\mathcal{A}} \beta_a \leq \sum_{a\in\mathcal{A}} \left[\theta_a \int_0^{x_a^{\mathrm{f}}(i)} t_a(\omega, \overline{y}_a)d\omega + (1-\theta_a)\int_0^{x_a^{\mathrm{f}}(i)} t_a(\omega, \underline{y}_a)d\omega\right] \quad \forall i \in \mathscr{V} \tag{29g}$$

$$y_a = \theta_a \overline{y}_a + (1-\theta_a)\underline{y}_a \quad \forall a \in \mathcal{A} \tag{29h}$$

$$0 \leq \theta_a \leq 1 \quad \forall a \in \mathcal{A} \tag{29i}$$

For correctness, we show that adding cuts (29b), (29f), and (29g) still gives us a lower bound on the optimal OFV of `CNDP`.

Proposition 2. *After adding cuts in the form of constraints (29b), (29f), and (29g), the optimal OFV of Formulation (29) is a lower bound on the optimal OFV of `CNDP`.*

Proof. By Proposition 1, adding the cut $B(\mathbf{x},\mathbf{y}) \leq B(\mathbf{x}^{\mathrm{f}},\mathbf{y})$ to obtain Formulation (11) yields a lower bound. Because of convexity of $B(\mathbf{x},\mathbf{y})$ by Assumption 1, the outer approximation of the LHS satisfies

$$\sum_{a\in\mathcal{A}} \beta_a \leq B(\mathbf{x},\mathbf{y}) \tag{30}$$

In addition, the convex combinations of the RHS of constraint (21e) satisfies

$$B\left(\mathbf{x}^{\mathrm{f}}, \mathbf{y}\right) \leq \sum_{a \in \mathcal{A}} \left[\theta_a \int_0^{x_a^{\mathrm{f}}(i)} t_a(\omega, \overline{y}_a) d\omega + (1-\theta_a) \int_0^{x_a^{\mathrm{f}}(i)} t_a(\omega, \underline{y}_a) d\omega \right] \tag{31}$$

Therefore, any $(\mathbf{x}, \mathbf{y})$ point that is feasible for Formulation (11), in particular constraint (11f), is also feasible for Formulation (29). Finally, constraint (29b) is the outer approximation of $Z(\mathbf{x}, \mathbf{y})$ (which is a convex function), and is therefore creates a lower bound on $Z(\mathbf{x}, \mathbf{y})$. □

The relationship between OA and the original convex nonlinear expressions was shown by Duran and Grossmann (1986):

Proposition 3. *As n^{oa} approaches infinity, $\sum_{a \in \mathcal{A}} \zeta_a$ approximates objective (12a) and $\sum_{a \in \mathcal{A}} \beta_a$ approximates $B(\mathbf{x}, \mathbf{y})$.*

The proof of Proposition 3 follows from Theorem 2 of Duran and Grossmann (1986). The number of points needed to obtain a sufficiently tight bound on objective (12a) was observed to not be too high in numerical results.

4.5 Spatial branch-and-cut algorithm for the CNDP

We present a spatial branch-and-cut algorithm for CNDP which is summarized in Algorithm 1. We denote $\mathcal{B}$ the set of branch nodes, which is initialized to contain the root node with no spatial branching in line 2. (The double bracket in line 2 indicates that $\mathcal{B}$ is initialized to a set containing the root node, which is itself a set of constraints.) Each iteration, we choose a branch node of the search tree and solve the Formulation (29) which is a linear program under Assumption 1 to obtain a lower bound for that subproblem. If we are not sufficiently close to the follower best response (line 14), then we choose one of the links a and divide its interval into smaller pieces. With sufficient value function cuts, and improved tightness from spatial branching, we show that our algorithm will converge to a global ϵ-optimal solution.

Line 13 updates the upper bound if the follower solution $Z(\mathbf{x}^{\mathrm{f}}, \mathbf{y}^{\mathrm{l}})$ is better than the best upper bound found thus far. Line 21 does the same with $Z(\mathbf{x}^{\mathrm{l}}, \mathbf{y}^{\mathrm{l}})$ (the leader solution) if it is within ϵ of the best follower solution found by solving $\texttt{TAP}(\mathbf{y}^{\mathrm{l}})$. Line 21 represents a special case: where Formulation (29) obtains a bi-level feasible solution on its own. When a converging algorithm like paired alternative segments is used to solve traffic assignment, the Beckmann value $B(\mathbf{x}^{\mathrm{f}}(i), \mathbf{y}^{\mathrm{l}}(i))$ may not represent the minimum Beckmann value possible. Therefore, we used the dual of traffic assignment to obtain a lower bound on the minimum Beckmann value to correctly check bi-level feasibility (Sugishita et al., 2025).

4.5.1 Convergence to ϵ-optimality

Algorithm 1 always has correct lower bounds on CNDP (Proposition 2), and its upper bounds come from feasible solutions to CNDP evaluated by TAP. For a given $\epsilon > 0$, Algorithm 1 converges because with enough branches, value function and OA cuts, the solution from

Algorithm 1: Spatial branching with cutting planes algorithm for `CNDP`

1 Set $LB \leftarrow 0$, $UB \leftarrow \infty$, $i \leftarrow 1$, $n^{\text{oa}} \leftarrow 0$, $n^{\text{vf}} \leftarrow 0$, $n^{\text{B}} \leftarrow 0$.
2 Set $\mathcal{B} = \{\{\mathbf{0} \leq \mathbf{y} \leq \overline{\mathbf{y}} \text{ from input}\}\}$.
3 **while** $UB - LB > \epsilon$ **do**
4 Remove a branch node b from $\mathcal{B}$.
5 Solve problem (29) on the $\mathbf{y}$ intervals specified by b to obtain $Z_{\text{lb}}(b)$, $\mathbf{x}^{\text{l}}(i)$, $\mathbf{y}^{\text{l}}(i)$.
6 **if** $Z_{\text{lb}}(b) < UB$ **then**
7 **for** $a \in \mathcal{A}$ **do**
8 Add cut (29b) on ζ_a using $(x_a^{\text{l}}(i), y_a^{\text{l}}(i))$.
9 Add cut (29f) on β_a using $(x_a^{\text{l}}(i), y_a^{\text{l}}(i))$.
10 Set $n^{\text{oa}} \leftarrow n^{\text{oa}} + 1$.
11 $LB \leftarrow \min_{b \in \mathcal{B}} Z_{\text{lb}}(b)$.
12 Solve `TAP` $\left(\mathbf{y}^{\text{l}}(i)\right)$ to obtain $\mathbf{x}^{\text{f}}(i)$.
13 Set $UB \leftarrow \min\left\{UB, Z\left(\mathbf{x}^{\text{f}}, \mathbf{y}^{\text{l}}\right)\right\}$.
14 **if** $B(\mathbf{x}^{\text{l}}(i), \mathbf{y}^{\text{l}}(i)) - B(\mathbf{x}^{\text{f}}(i), \mathbf{y}^{\text{l}}(i)) > \epsilon$ **then**
15 Add RHS of value function cut (29g) using $\mathbf{x}^{\text{f}}(i)$. Set $n^{\text{vf}} \leftarrow n^{\text{vf}} + 1$.
16 Choose a for spatial branching.
17 Calculate branch split $\tilde{y}_a$ via Eq. (20).
18 $b^{\text{left}} \leftarrow b \cup \{y_a \leq \tilde{y}_a\}$. $\mathcal{B} \leftarrow \mathcal{B} \cup \{b^{\text{left}}\}$.
19 $b^{\text{right}} \leftarrow b \cup \{y_a \geq \tilde{y}_a\}$. $\mathcal{B} \leftarrow \mathcal{B} \cup \{b^{\text{right}}\}$.
20 **else**
21 Set $UB \leftarrow \min\left\{UB, Z\left(\mathbf{x}^{\text{l}}, \mathbf{y}^{\text{l}}\right)\right\}$.
22 Set $i \leftarrow i + 1$.

Formulation (29) will be within ϵ of being bi-level feasible, i.e., a solution to `TAP`. At that point, further branching becomes unnecessary and that node in the search tree can be fathomed. Proposition 4 proves that sufficient (discrete) spatial branches ensure that the solution to Formulation (29) is within ϵ of optimal OFV of `CNDP`.

Proposition 4. *For a given $\epsilon > 0$, with sufficiently tight bounds $\underline{\mathbf{y}}$ and $\overline{\mathbf{y}}$, sufficient outer approximation cuts (29b), and sufficient value function cuts (29f), the solution $(\mathbf{x}^{\text{l}}, \mathbf{y}^{\text{l}})$ to Formulation (29) will satisfy $B(\mathbf{x}^{\text{l}}, \mathbf{y}^{\text{l}}) - \min_{\mathbf{x}} B(\mathbf{x}, \mathbf{y}^{\text{l}}) \leq \epsilon$. Furthermore, for a given $\epsilon > 0$, Algorithm 1 will return a point $(\mathbf{x}^{\star}, \mathbf{y}^{\star})$ corresponding to the best upper bound UB found during search such that the gap between the global lower bound LB on the optimal OFV of `CNDP` will satisfy $Z(\mathbf{x}^{\star}, \mathbf{y}^{\star}) - LB \leq \epsilon$.*

Proof. For each branch node b evaluated by Algorithm 1, there are three cases for the output of the evaluation. Case 1: it is possible that $Z_{\text{lb}}(b) \geq UB$, and the branch can be pruned in line 6 because it will never lead to a be better feasible solution than the best upper bound. Case 2: the $(\mathbf{x}^{\text{l}}, \mathbf{y}^{\text{l}})$ solution to Formulation (29) may be within ϵ of `TAP`$(\mathbf{y}^{\text{l}})$, and no further branching is needed (line 14). Case 3: the follower best response is violated, and spatial

branching continues. In this case, spatial branching will eventually terminate in cases 1 or 2 by ϵ-optimality (Liberti, 2008; Floudas, 2013). □

4.5.2 Implementation discussion

Convergence of Algorithm 1 occurs as shown by Proposition 4. However, convergence can be limited if traffic assignment is not solved to sufficient precision. If so, the Beckmann value $B(\mathbf{x}, \mathbf{y})$ used in cut (29g) will fail to exclude enough $\mathbf{x}$ values from Formulation (29) resulting in lack of convergence. Practically speaking, traffic assignment will be solved to some level of precision, but not to exact optimality. The level of precision determines the range of $\mathbf{x}$ values that are excluded by value functions using $\mathbf{x}^{\text{f}} = \texttt{TAP}(\mathbf{y})$. The cut is always valid because $\texttt{TAP}(\mathbf{y})$ seeks to minimize $B(\mathbf{x}, \mathbf{y})$, but the cut may not be efficient unless `TAP` is solved to sufficiently high precision. With dedicated, efficient traffic assignment algorithms, it is possible to quickly solve traffic assignment to a high precision, e.g., $1e^{-12}$ (e.g. Bar-Gera, 2010) which is expected to be much lower than the precision expected for value function cuts (ϵ) and for solving `CNDP` globally (ϵ).

The largest issue with convergence comes from proving optimality, i.e., increasing the lower bound found by the augmented high-point relaxation to be within ϵ of the best solution. Therefore, the choice of branch node in line 4 is typically the node with the lowest lower bound. One potential advantage comes from having a small number of y_a variables relative to the number of links. For certain networks and scenarios, it is possible for Algorithm 1 to converge without spatial branching simply by adding outer approximation cuts of (29f) and value function cuts (29g) without changing the bounds on $\mathbf{y}$.

Algorithm 1 involves solving a sequence of linear programs (29) with linear OA cuts in line 5, and user equilibrium traffic assignment for fixed $\mathbf{y}$ in line 13. Commercial solvers can solve linear programs efficiently, and dedicated algorithms can solve traffic assignment efficiently on large networks (e.g. Bar-Gera, 2010). We will use column generation to further speed up the solution of Formulation (29). Although many iterations may be required, each iteration should be relatively fast.

4.6 Column generation and spatial branch-and-price-and-cut algorithm

In Formulation (29), we explicitly write out path flow variables h^{π}. Algorithms requiring enumerating all paths $\pi \in \Pi$ will not scale well to larger networks, and we therefore use column generation to avoid path enumeration. Some prior globally optimal solution methods for `CNDP` enumerated all paths (Wang and Lo, 2010; Du and Wang, 2016) because they used complementary slackness to enforce follower optimality and needed to ensure that complementary slackness holds for all paths. Our approach admits considering only a partial set of paths through column generation.

First, we replace the h^{π} variables with origin-based link flows. Let z_a^r be the flow from

origin r on link a. Then constraints (29c) and (29d) are replaced with

$$x_{ij} = \sum_{r \in \mathcal{Z}} z_{ij}^r \qquad \forall (i,j) \in \mathcal{A} \tag{32a}$$

$$\sum_{i \in \text{Inc}(j)} z_{ij}^r - \sum_{k \in \text{Out}(j)} z_{jk}^r = \begin{cases} \sum_{s \in \mathcal{Z}} d_{rs} & \text{if } j = r \\ -d_{rj} & \text{if } j \in \mathcal{Z} \\ 0 & \text{else} \end{cases} \qquad \forall j \in \mathcal{N}, \forall r \in \mathcal{Z} \tag{32b}$$

$$z_a^r \geq 0 \qquad \forall a \in \mathcal{A}, \forall r \in \mathcal{Z} \tag{32c}$$

where $\text{Inc}(j)$ and $\text{Out}(j)$ are the sets of incoming and outgoing links to node j, respectively. Similarly, Luathep et al. (2011) replaced the path-based complementary constraints in Wang and Lo (2010) with link-based versions. However, this approach involves multi-commodity link flow variables, which is still not efficient for large networks.

Since h^π only appears in two constraints in Formulation (29), implementing column generation is fairly easy despite the sequential addition of value function and OA cuts. Let ψ_a be the dual variable of constraint (29c), and let σ_{rs} be the dual variable of constraint (29d). The reduced cost of h^π, denoted c^π, is

$$c^\pi = -\sigma_{rs} + \sum_{a \in \mathcal{A}} \delta_a^\pi \psi_a \tag{33}$$

Since the goal is to minimize congestion, larger link flows increase the objective. Therefore, we can rewrite constraint (29c) as

$$x_a - \sum_{a \in \mathcal{A}} \delta_a^\pi h^\pi \geq 0 \tag{34}$$

which restricts the sign of ψ_a to $\psi_a \geq 0$ (Lübbecke and Desrosiers, 2005). Similarly, we can rewrite constraint (29d) as

$$\sum_{\pi \in \Pi_{rs}} h^\pi \geq d_{rs} \tag{35}$$

which restricts the sign of σ_{rs} to $\sigma_{rs} \geq 0$.

If there are any paths where $c^\pi < 0$, adding corresponding path flow variables could reduce the objective value of Formulation (29). We can search for such paths using a shortest path algorithm on the dual link costs ψ_a. Because $\psi_a \geq 0$ after rewriting Eq. (29c) as Eq. (34), all link costs for this shortest path calculation are non-negative. The column generation method for solving Formulation (29) in line 5 of Algorithm 1 is given in Algorithm 2.

5 Numerical results

The purpose of the numerical results is to demonstrate that our spatial branch-and-price-and-cut algorithm solves `CNDP` to ϵ-global optimality for a predefined $\epsilon > 0$, and does so

Algorithm 2: Column generation for solving Formulation (29) (line 5 of Algorithm 1)

repeat
  Solve Formulation (29) on the restricted path set $\tilde{\Pi}$ to obtain link duals ψ_a and origin-destination duals σ_{rs}.
  for $(r, s) \in \mathcal{Z}^2$ **do**
    $\pi^\star_{rs} \leftarrow \arg\min_{\pi \in \Pi_{rs}} \left\{ \sum_{a \in \mathcal{A}} \delta^\pi_a \psi_a \right\}$ (shortest path)
    $c^{\pi^\star_{rs}} \leftarrow -\sigma_{rs} + \sum_{a \in \mathcal{A}} \delta^{\pi^\star_{rs}}_a \psi_a$
    if $c^{\pi^\star_{rs}} < 0$ **then**
      Add $\pi^\star_{rs}$ to the restricted path set $\tilde{\Pi}$.
until $\min_{(r,s) \in \mathcal{Z}^2} c^{\pi^\star_{rs}} \geq 0$.

faster than the alternatives in the literature. Towards that goal, we examine several recent algorithms to find globally optimal solutions to CNDP. Wang and Lo (2010) formulated CNDP as a MILP using complementary slackness to maintain follower optimality and a piecewise-linear approximation of the nonlinear travel times $t_a(x_a, y_a)$. Due to the use of complementary slackness, they required path enumeration and binary variables per path. Du and Wang (2016) reformulated the single-level problem of Wang and Lo (2010) (using complementary slackness for follower optimality) using geometric programming. While geometric programming can be solved relatively quickly, their approach also used path enumeration, and it is not clear how to use column generation within a geometric programming framework. Furthermore, although geometric programming handles complementary slackness well, it requires constructing approximations to linear constraints such as (6b) and (6c), which is inefficient. Li et al. (2012) avoided path enumeration by solving a sequence of concave minimization problems (minimizing a concave function over a convex set). Concave minimization is an NP-hard problem, and they solved it by enumerating over the extreme points of the feasible region, which is also computationally intensive. Overall, all prior globally optimal solution algorithms for CNDP require solving difficult subproblems. Consequently, Wang and Lo (2010), Li et al. (2012), and Du and Wang (2016) all demonstrated their algorithm on the small Harker and Friesz (1984) test network (Figure 1) with 6 nodes and 16 links.

There are also several recent algorithms proven to find stationary points (local minima) to CNDP (Wang et al., 2022; Guo et al., 2025). Our goal is to find a solution that is proven to be within ϵ of global optimality. Algorithms to find a local minima are likely to be faster but cannot guarantee that the solution found is a global optimum. Therefore, stationary point algorithms are not directly comparable to our BPC.

Our spatial branch-and-price-and-cut algorithm benefits from solving two relatively fast problems: a linear program using column generation, and standard traffic assignment. We aim to show a substantial reduction in computation time on the Harker and Friesz (1984) network, and the ability to solve CNDP to global optimality on much larger networks. For

reproducibility, all data and codes are made available at [*public repository url will be provided after peer-review*].

- We implemented Wang and Lo (2010)'s MILP. Because it uses a piecewise-linear approximation, we calculated the objective value as $Z(\texttt{TAP}(\mathbf{y}^\star), \mathbf{y}^\star)$ after obtaining $\mathbf{y}^\star$ from solving the MILP.

- We studied the algorithm of Du and Wang (2016), but the geometric programming reformulation of CNDP was not actually stated in Du and Wang (2016) and we were not able to program an unformulated problem into CPlex. They provide a computational comparison with Wang and Lo (2010) on the Harker and Friesz (1984) network, but their computation times on the 16-link Harker and Friesz (1984) network are still 1 min or more, which is orders of magnitude greater than our computation time on that network. Furthermore, they require path enumeration, which will not scale to large networks, and it is therefore not clear whether they could solve their problem on larger networks.

- We started implementing Li et al. (2012)'s algorithm. However, on the Harker and Friesz (1984) network, it required 2997.7 sec to enumerate the extreme points using the `polytope` Python package, and we were unable to enumerate the extreme points on larger networks like Sioux Falls within a reasonable duration. Since enumerating the extreme points is a subproblem of the concave minimization subproblem of their method, and has to be solved multiple times, their method is clearly not competitive in terms of computation time. We note that Li et al. (2012) reported requiring 12hrs to solve CNDP on the Harker and Friesz (1984) network.

Overall, the reported results in the literature indicate that solving CNDP on Harker-Friesz in less than 1 second (Table 1) is a massive improvement over existing algorithms. Moreover, the literature on globally optimal CNDP algorithms is highly limited.

All algorithms were implemented in Python 3 using IBM's CPlex solver for solving linear programs and MILPs. Traffic assignment by paired alternative segments (Bar-Gera, 2010) was used to solve traffic assignment. Reported computation times are from a Mac mini desktop with a M4 processor. We assumed that $g_a(y_a)$ were linear functions $\gamma_a \cdot y_a$ where γ_a is the cost per capacity increase, and used the BPR travel time function with parameters from published test networks.

5.1 CNDP on Harker and Friesz (1984) network

Network parameters for this network were published in Harker and Friesz (1984). Table 1 compares computation times of our spatial branch-and-price-and-cut algorithm and the MILP of Wang and Lo (2010). We scaled the link costs γ_a and the total trips to create different scenarios. The number of pieces parameter for the MILP indicates how many different bounds were used for x_a and y_a in the piecewise linear approximation for travel time. We terminated both the MILP and our algorithm at a 1% optimality gap, meaning

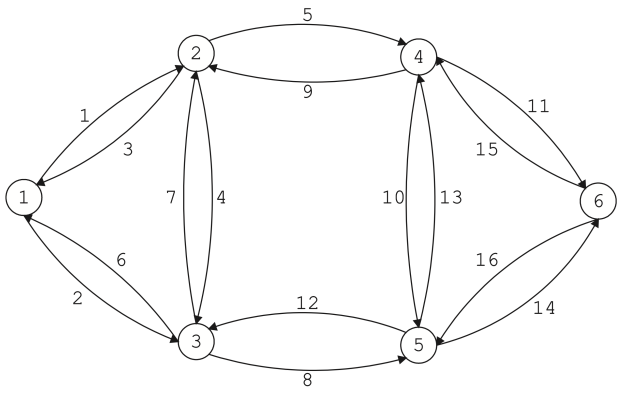

Figure 1: Harker-Friesz test network (reproduced from Li et al., 2012)

that we found a solution that was proven to be within 1% of global optimality. For traffic assignment, we required the solution to be within 0.1% of the optimal Beckmann value to consider it to be bi-level feasible. The MILP was terminated after a 4hr time limit; in many cases, the optimality gap was large, or no feasible solution was found. The MILP solution differs slightly from the OA solution due to the piecewise linear approximation, but overall the differences were not large when the MILP terminated with low optimality gap.

Table 1: Results from Wang and Lo (2010)'s MILP and our spatial branch-and-price-and-cut algorithm on the Harker and Friesz (1984) network.

demand scale	cost scale	Wang and Lo (2010)'s MILP						Our BPC algorithm			
		5 piece linear approximation			7 piece linear approximation						
		obj	rt (s)	gap	obj	rt (s)	gap	obj	rt(s)	gap	iter
0.25	1.0	56.5	10964.99	0.01	inf	14403.03†		56.5	0.03	0.0	1§
0.5	1.0	116.8	7583.24	0.01	inf	14403.46†		116.8	0.03	0.0	3
0.75	1.0	186.6	14403.48†	0.704	inf	14422.58†		186.7	0.03	0.0	4
1.0	1.0	inf	14403.46†		278.6	14415.87†	0.897	278.7	0.04	0.0	4
1	2.0	557.5	14400.8†	0.805	inf	14403.95†		286.7	0.03	0.0	4
1	4.0	1114.9	14402.73†	0.66	1114.4	14403.71†	0.735	298.4	0.04	0.0	7
1	8.0	inf	14403.67†		2228.8	14416.4†	0.878	315.3	0.04	0.0	6

*Indicates termination without CPlex reporting a feasible solution
†Indicates termination due to 4hr time limit
§Indicates termination without spatial branching

The MILP is much slower than our algorithm in all instances. Furthermore, improving the piecewise-linear approximation with more pieces tends to increase computation time. This pattern is consistent with results published by Du and Wang (2016): for some scenarios, they obtained fast results for the MILP, but for others, the MILP was very slow. Although Du and Wang (2016) reported better computation times for their geometric programming approach, their approach requires enumerating paths and is not expected to scale to larger networks. On all instances, the proposed algorithm can solve CNDP on this network much faster than the MILP approach, i.e., less than 1 second compared to 4 hours of computing time.

5.2 Globally optimal solutions to CNDP on larger networks

The main benefit the proposed spatial branch-and-price-and-cut algorithm has over published work is the potential to solve CNDP on larger networks than the Harker and Friesz (1984) test network (Figure 1), which was the largest network that other algorithms could be demonstrated on (Wang and Lo, 2010; Li et al., 2012; Du and Wang, 2016). We report globally optimal solutions on networks with up to 914 links, which has not previously been achieved in the literature on CNDP. The benefits of our algorithm is how it iteratively uses two relatively simple subproblems — solving a linear program, and solving traffic assignment. Both these subproblems can be solved quickly. We terminated our algorithm at a 1% optimality gap, meaning that we found a feasible solution that was proven to be within 1% of global optimality. For bi-level feasibility, we required the solution to be within 0.1% of the optimal Beckmann value from traffic assignment. We used the dual of traffic assignment (Sugishita et al., 2025) to compute a lower bound on the optimal Beckmann value to check bi-level feasibility.

Overall, the results in Tables 2 and 3 are on networks far larger than the Harker and Friesz (1984) test network used in prior work, and we cannot even run prior algorithms (Li et al., 2012; Du and Wang, 2016) on networks of this size to compare solutions. Recent stationary point algorithms (Wang et al., 2022; Guo et al., 2025) can be run on such large networks, but cannot guarantee that the solution found is within ϵ of global optimality.

To demonstrate that our method robustly solves CNDP instances based on these networks, we consider different instances, shown in the "Inst." column of Tables 2 and 3, with different randomly generated costs and different links where $y_a \geq 0$ is permitted. We vary the number of y_a variables that are permitted to be non-zero. As the number of non-zero y_a variables increases, the congestion in the network generally decreases, therefore instances where $y_a > 0$ is permitted for all links become less interesting.

For example, compare the total system travel times in Table 2 for instances on the same network with different numbers of non-zero y_a variables. Costs per capacity increase are randomly generated because they are typically not published with standard traffic assignment network data unless they are being used for CNDP. y_a was limited to $C_a/2$, for a maximum of a 50% increase in link capacity. (The average value of the link cost is reported in the "avg. cost" column in units of $ per additional veh/hr capacity. We also varied the number of y_a variables that could be non-negative, shown in the "y_a vars" column of Tables 2 and

3, the average link cost, and the demand.

For each instance, we report the objective, gap at termination, and total system travel time (which excludes the cost of link capacity additions from the objective). The difference between the objective $Z(\mathbf{x}, \mathbf{y})$ and total system travel time indicates when $\mathbf{y} \geq \mathbf{0}$ in the optimal solution. We also show the total system travel time with 0 additional capacity for comparison. The objective and total system travel time are from the best upper bound, i.e. the best `TAP` solution for the $\mathbf{y}$ obtained from Formulation (29), so they are guaranteed to be follower-optimal for CNDP. We terminated when the gap was below 1%. We also report the total computation time ("Tot. time"), time spent on solving the traffic assignment subproblem ("`TAP` time"), and the number of iterations ("iter.") of Algorithm 1 before termination.

Note that the number of iterations refers to the number of spatial branching nodes solved. For some instances, Algorithm 1 terminated without spatial branching due to the low gap. We added value function cuts to the root node to improve its lower bound, and in some instances could achieve 1% gap without tightening the RHS of constraint (29g) with spatial branching. Sioux Falls was observed to have some of the most difficult instances to solve in terms of the number of iterations required for convergence. The cause of the large number of iterations is the difficulty in finding solutions that have a sufficiently small follower objective value to be bi-level feasible. The difficulty of Sioux Falls instances may be partially due to the relatively small number of links on Sioux Falls. For a larger network like Anaheim, 60 y_a variables is only 6.5% of the links, so much of the numerical value of the value function cut (29g) does not vary with y_a. In contrast, 20 y_a variables on Sioux Falls is 26.3% of the links.

We also report the average capacity added to links where $y_a > 0$ is permitted as a percentage of the current capacity, i.e. the average value of y_a/C_a. The possible value of y_a/C_a in our instances ranges from 0 to 0.5. The values obtained indicate that the optimal solution is not choosing extreme y_a values, but something in between, which is consistent with the non-linearity of the problem. Furthermore, if we increase the average cost, there is a clear decrease in the average y_a/C_a, indicating the importance of the costs for capacity increase.

Computation time per iteration varies. During later iterations, column generation to solve the augmented high-point relaxation is faster because the restricted path set has been populated by prior iterations. In addition, the $\mathbf{y}^{\text{l}}$ is often very similar between iterations, to the point where the traffic assignment subproblem does not always need to be solved again. The reported solution comes from the best follower-optimal solution to CNDP by solving $\texttt{TAP}(\mathbf{y}^{\text{l}})$ using the $\mathbf{y}^{\text{l}}$ from Formulation (29), so it is a valid solution regardless of whether `TAP` is solved again. Instead, the main work is in proving optimality by obtaining a tighter (more correct) lower bound from Formulation (29).

Table 2: Performance of the spatial branch-and-price-and-cut algorithm on larger networks.

Network	Non-zero y_a vars	Inst.	Avg. cost	Obj.	Avg. y_a/C_a	gap	TSTT	Tot. time	`TAP` time	iter.
Sioux Falls	10	1	1.79e-01	6859.7	0.323	0.27%	6859.7	1.99s	1.27s	6
		2	2.85e-01	6864.2	0.354	0.00%	6864.2	2.63s	1.85s	7
24 nodes		3	2.13e-01	6693.6	0.380	0.00%	6693.6	2.44s	1.73s	5
76 links		1	1.79e+01	7157.0	0.217	0.07%	7157.0	2.54s	1.44s	14
24 zones		2	2.85e+01	7133.3	0.128	0.11%	7133.3	3.00s	2.34s	4
		3	2.13e+01	7023.1	0.165	0.00%	7023.1	1.99s	1.29s	5
360,600 trips	20	1	2.89e-01	6399.7	0.312	0.00%	6399.7	3.49s	2.54s	9
		2	2.53e-01	6237.5	0.387	0.10%	6237.5	5.13s	3.47s	21
		3	2.50e-01	6262.9	0.315	0.31%	6262.9	4.01s	2.74s	14
		1	2.89e+01	6930.5	0.105	0.07%	6930.5	8.77s	4.62s	51
		2	2.53e+01	6847.6	0.147	0.16%	6847.6	13.32s	7.27s	67
		3	2.50e+01	6866.6	0.149	0.05%	6866.6	15.52s	10.00s	60
	0			7475.7			7475.7			
Eastern-Massachusetts	10	1	9.22e-03	2055.8	0.345	0.75%	2055.8	60.19s	47.31s	1[§]
		2	7.94e-03	1743.6	0.350	0.99%	1743.6	66.39s	50.01s	1[§]
74 nodes		3	9.29e-03	1878.6	0.410	0.80%	1878.6	66.63s	53.98s	1[§]
248 links		1	9.22e-01	2058.0	0.104	0.77%	2058.0	58.70s	47.03s	1[§]
74 zones		2	7.94e-01	1749.7	0.172	0.79%	1749.7	62.32s	48.16s	1[§]
		3	9.29e-01	1886.2	0.211	0.54%	1886.2	68.00s	54.65s	1[§]
262,306 trips	30	1	9.47e-03	1282.7	0.339	0.76%	1282.7	88.92s	75.23s	1[§]
		2	9.89e-03	1600.8	0.275	0.92%	1600.8	99.43s	85.52s	1[§]
		3	9.36e-03	1904.7	0.436	0.64%	1904.7	92.81s	80.15s	1[§]
		1	9.47e-01	1299.3	0.210	0.74%	1299.3	90.46s	78.07s	1[§]
		2	9.89e-01	1610.8	0.089	0.97%	1610.8	70.64s	55.63s	1[§]
		3	9.36e-01	1920.9	0.150	0.66%	1920.9	95.21s	83.09s	1[§]
	0			2063.7			2063.7			

[†]Indicates termination due to time limit

[§]Indicates termination without spatial branching

Table 3: Spatial branch-and-price-and-cut algorithm performance.

Network	Non-zero y_a vars	Inst.	Avg. cost	Obj.	Avg. y_a/C_a	gap	TSTT	Tot. time	`TAP` time	iter.
Berlin Mitte Center	30	1	1.47e-01	12323.0	0.163	0.93%	12323.0	62.22s	24.70s	1[§]
		2	1.41e-01	11827.7	0.205	0.04%	11827.7	153.11s	42.90s	13
398 nodes		3	1.70e-01	12142.4	0.217	0.93%	12142.4	76.70s	26.65s	3
871 links		1	1.47e+00	12331.6	0.129	0.99%	12331.6	61.77s	24.72s	1[§]
36 zones		2	1.41e+00	11955.0	0.218	0.97%	11955.0	175.24s	48.74s	15
		3	1.70e+00	12155.9	0.142	0.99%	12155.9	84.69s	28.74s	3
22,963 trips	60	1	1.40e-01	12102.6	0.207	0.98%	12102.6	110.85s	31.33s	7
		2	1.45e-01	12045.2	0.188	0.95%	12045.2	109.78s	33.34s	5
		3	1.38e-01	11872.4	0.277	0.97%	11872.4	146.07s	35.30s	9
		1	1.40e+00	12123.8	0.152	0.98%	12123.8	134.10s	36.71s	13
		2	1.45e+00	12059.0	0.155	0.99%	12059.0	108.50s	34.32s	5
		3	1.38e+00	11897.2	0.191	0.99%	11897.2	142.10s	40.09s	15
	0			12454.1			12454.1			
Anaheim	30	1	2.18e-01	101757.5	0.238	0.84%	101757.5	209.73s	28.86s	1[§]
		2	2.52e-01	101313.4	0.310	0.88%	101313.4	203.47s	28.85s	1[§]
416 nodes		3	2.77e-01	101354.3	0.289	0.95%	101354.3	200.49s	29.03s	1[§]
914 links		1	2.18e+01	101848.6	0.033	0.95%	101848.6	203.65s	28.11s	1[§]
38 zones		2	2.52e+01	101732.7	0.039	0.95%	101732.7	214.44s	28.13s	1[§]
		3	2.77e+01	101728.0	0.045	0.95%	101728.0	230.66s	28.05s	1[§]
418,778 trips	60	1	2.28e-01	95756.2	0.275	0.96%	95756.2	203.83s	32.20s	1[§]
		2	2.28e-01	100256.4	0.249	0.86%	100256.4	192.91s	33.91s	1[§]
		3	2.38e-01	100660.6	0.286	0.79%	100660.6	225.45s	32.14s	1[§]
		1	2.28e+01	96186.3	0.052	0.92%	96186.3	193.41s	30.66s	1[§]
		2	2.28e+01	100872.6	0.062	0.86%	100872.6	228.25s	29.36s	1[§]
		3	2.38e+01	101230.6	0.059	0.94%	101230.6	214.35s	30.31s	1[§]
	0			101889.2			101889.2			

[†]Indicates termination due to time limit

[§]Indicates termination without spatial branching

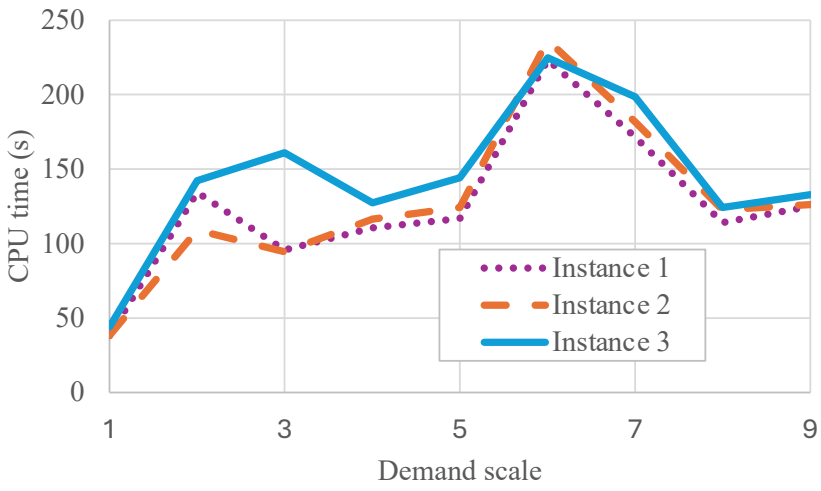

Figure 2: Comparison of computation time against demand scaling on the Berlin Mitte Center network with 60 non-zero y_a variables. The 3 instances (with randomly chosen candidate links) correspond to the Berlin-Mitte-Center instances in Table 3.

5.2.1 Effect of changing demand on computation time

Higher demand could make the BPC less effective for two reasons. First, higher demand makes traffic assignment more difficult to solve (relative to a network with relatively low congestion). Second, the travel time function $t_a(x_a, y_a)$ becomes less linear as the volume-to-capacity ratio increases, which makes the outer approximation less tight. Figure 2 shows the relationship between computation time and demand on Berlin Mitte Center. As expected, as demand increases, computation time starts to increase. However, at high levels of demand, computation time started to decrease again. This is likely because at high demand levels, the optimal solution is to choose the maximum capacity increase of Y_a.

5.2.2 Effect of changing ϵ on computation time

Table 3 reports computation times for 1%-global optimality (i.e. solutions are within 1% of global optimality). To study the effect of gap on computation time, we solved Berlin-Mitte-Center instances using 0.1% gap as the stopping criteria. The results are reported in Table 4. As expected, additional computation time and iterations are required to reach the desired gap. From this, we conclude that our BPC can effectively solve instances to a smaller optimality gap if desired.

Table 4: Effect of 0.1% optimality gap on computational results for Berlin-Mitte-Center

Non-zero y_a vars	Inst.	Obj.	Avg. y_a/C_a	gap	TSTT	Tot. time	iter.
60	1	12016.9	0.117	0.04%	12016.9	470.8 s	25
	2	11957.1	0.106	0.07%	11957.1	274.7 s	9
	3	11794.9	0.154	0.02%	11794.9	596.6 s	47

[§]Indicates termination without spatial branching

6 Conclusions

The CNDP has been extensively studied in the literature, but existing methods for finding globally optimal solutions to the CNDP require solving difficult problems creating large computation times for small test networks. Recent work has primarily focused on efficiently finding stationary points that are not guaranteed to be globally optimal (Wang et al., 2022; Guo et al., 2025). Consequently, there is a gap in the literature to create a computationally efficient algorithm for finding globally optimal solutions

Using value function cuts and outer approximation (OA), we created one such algorithm that sequentially solves linear programs and traffic assignment problems. Convergence occurs because when $\mathbf{x}$ from the high-point relaxation SO-CNDP is not follower-optimal, we add a value function cut based on the follower objective value to exclude $\mathbf{x}$ and other points like it from the SO-CNDP mathematical program. Since such cuts are valid at follower optimality, the revised SO-CNDP mathematical program yields a lower bound to CNDP, that is gradually tightened. We used OA to reformulate the nonlinear leader objective and value function cuts into a sequence of linear programs, which is valid when these nonlinear functions are convex (which they are for the Bureau of Public Roads travel time function). Separating the OA by link created a much stronger OA for faster convergence. Column generation was used to more quickly solve the network-based linear programming subproblem. Spatial branching is required in theory to convexify the value function constraints, but on larger networks it may not be required to achieve a desired level of convergence.

Due to the computational complexity of their algorithms, prior work on finding globally optimal solutions to CNDP required large computation times of 1 minute or more to solve CNDP on the small Harker and Friesz (1984) test network with 6 nodes and 16 links (Wang and Lo, 2010; Du and Wang, 2016). In contrast, we were able to obtain solutions within 1% of global optimality on networks with up to 416 nodes and 914 links in reasonable computation times. Our algorithm is faster because it relies on iteratively solving two simple subproblems — a linear program and traffic assignment — which can both be solved quickly. The ability to solve CNDP on problem instances based on networks that are two orders of magnitude larger than those in the literature. Furthermore, we believe that our algorithm is significantly easier to implement than methods from prior work as it relies on linear programs and traffic assignment.

However, there are several limitations that would be good to improve on in future work. From a computational standpoint, the SO-CNDP linear program remains difficult to solve

for larger networks, and requires to be solved repeatedly to achieve convergence. Methods to solve this subproblem faster would help for solving CNDP on larger networks. Furthermore, our algorithm requires convexity of both the leader objective and the value function cuts based on the follower objective. Although many network design problems are based on the Bureau of Public Roads travel time function and satisfy these assumptions, in other contexts these convexity assumptions are restrictive.

References

M. Abdulaal and L. J. LeBlanc. Continuous equilibrium network design models. *Transportation Research Part B: Methodological*, 13(1):19–32, 1979.

R. O. Arbex and C. B. da Cunha. Efficient transit network design and frequencies setting multi-objective optimization by alternating objective genetic algorithm. *Transportation Research Part B: Methodological*, 81:355–376, 2015.

S. Asadi Bagloee and M. Sarvi. An outer approximation method for the road network design problem. *Plos one*, 13(3):e0192454, 2018.

J. X. Ban, H. X. Liu, M. C. Ferris, and B. Ran. A general mpcc model and its solution algorithm for continuous network design problem. *Mathematical and Computer Modelling*, 43(5-6):493–505, 2006.

H. Bar-Gera. Traffic assignment by paired alternative segments. *Transportation Research Part B: Methodological*, 44(8-9):1022–1046, 2010.

M. Beckmann, C. B. McGuire, and C. B. Winsten. Studies in the economics of transportation. Technical report, 1956.

D. Braess. Über ein paradoxon aus der verkehrsplanung. *Unternehmensforschung*, 12(1): 258–268, 1968.

Z. Chen, F. He, and Y. Yin. Optimal deployment of charging lanes for electric vehicles in transportation networks. *Transportation Research Part B: Methodological*, 91:344–365, 2016.

Z. Chen, F. He, Y. Yin, and Y. Du. Optimal design of autonomous vehicle zones in transportation networks. *Transportation Research Part B: Methodological*, 99:44–61, 2017.

S.-W. Chiou. Bilevel programming for the continuous transport network design problem. *Transportation Research Part B: Methodological*, 39(4):361–383, 2005.

S.-W. Chiou. A generalized iterative scheme for network design problem. *Applied Mathematics and Computation*, 188(2):1115–1123, 2007.

S.-W. Chiou. A subgradient optimization model for continuous road network design problem. *Applied Mathematical Modelling*, 33(3):1386–1396, 2009.

R. B. Dial. A path-based user-equilibrium traffic assignment algorithm that obviates path storage and enumeration. *Transportation Research Part B: Methodological*, 40(10):917–936, 2006.

B. Du and D. Z. Wang. Solving continuous network design problem with generalized geometric programming approach. *Transportation Research Record*, 2567(1):38–46, 2016.

M. A. Duran and I. E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, 36(3):307–339, 1986.

R. Z. Farahani, E. Miandoabchi, W. Y. Szeto, and H. Rashidi. A review of urban transportation network design problems. *European Journal of Operational Research*, 229(2): 281–302, 2013.

H. Farvaresh and M. M. Sepehri. A single-level mixed integer linear formulation for a bi-level discrete network design problem. *Transportation Research Part E: Logistics and Transportation Review*, 47(5):623–640, 2011.

H. Farvaresh and M. M. Sepehri. A branch and bound algorithm for bi-level discrete network design problem. *Networks and Spatial Economics*, 13(1):67–106, 2013.

C. A. Floudas. *Deterministic global optimization: theory, methods and applications*, volume 37. Springer Science & Business Media, 2013.

P. Fontaine and S. Minner. Benders decomposition for discrete–continuous linear bilevel problems with application to traffic network design. *Transportation Research Part B: Methodological*, 70:163–172, 2014.

M. Gairing, T. Harks, and M. Klimm. Complexity and approximation of the continuous network design problem. *SIAM Journal on Optimization*, 27(3):1554–1582, 2017.

Z. Gao, J. Wu, and H. Sun. Solution algorithm for the bi-level discrete network design problem. *Transportation Research Part B: Methodological*, 39(6):479–495, 2005.

Z. Gao, H. Sun, and H. Zhang. A globally convergent algorithm for transportation continuous network design problem. *Optimization and Engineering*, 8:241–257, 2007.

L. Guo, W. Zhou, X. Wang, H. Yang, and T. Fan. Penalty decomposition methods for second-best congestion pricing problems on large-scale networks. *INFORMS Journal on Computing*, 2024.

L. Guo, H. Yin, and J. Zhang. A penalized sequential convex programming approach for continuous network design problems. *INFORMS Journal on Computing*, 2025.

P. T. Harker and T. L. Friesz. Bounding the solution of the continous equilibrium network design problem. In *Papers presented during the Ninth International Symposium on Transportation and Traffic Theory held in Delft the Netherlands, 11-13 July 1984.*, 1984.

L. J. Leblanc. An algorithm for the discrete network design problem. *Transportation Science*, 9(3):183–199, 1975.

M. W. Levin, E. Wong, B. Nault-Maurer, and A. Khani. Parking infrastructure design for repositioning autonomous vehicles. *Transportation Research Part C: Emerging Technologies*, 120:102838, 2020.

C. Li, H. Yang, D. Zhu, and Q. Meng. A global optimization method for continuous network design problems. *Transportation Research Part B: Methodological*, 46(9):1144–1158, 2012.

L. Liberti. Introduction to global optimization. *Ecole Polytechnique*, 1:1–43, 2008.

P. Luathep, A. Sumalee, W. H. Lam, Z.-C. Li, and H. K. Lo. Global optimization method for mixed transportation network design problem: a mixed-integer linear programming approach. *Transportation Research Part B: Methodological*, 45(5):808–827, 2011.

M. E. Lübbecke and J. Desrosiers. Selected topics in column generation. *Operations Research*, 53(6):1007–1023, 2005.

P. Marcotte. Network design problem with congestion effects: A case of bilevel programming. *Mathematical Programming*, 34(2):142–162, 1986.

Q. Meng, H. Yang, and M. G. Bell. An equivalent continuously differentiable model and a locally convergent algorithm for the continuous network design problem. *Transportation Research Part B: Methodological*, 35(1):83–105, 2001.

A. Mirheli and L. Hajibabai. Charging network design and service pricing for electric vehicles with user-equilibrium decisions. *IEEE Transactions on Intelligent Transportation Systems*, 24(3):2888–2902, 2023.

M. A. Nayeem, M. K. Rahman, and M. S. Rahman. Transit network design by genetic algorithm with elitism. *Transportation Research Part C: Emerging Technologies*, 46:30–45, 2014.

D. Rey. Computational benchmarking of exact methods for the bilevel discrete network design problem. *Transportation Research Procedia*, 47:11–18, 2020.

D. Rey and M. W. Levin. A branch-and-price-and-cut algorithm for discrete network design problems under traffic equilibrium. *Optimization online*, 2024. URL https://optimization-online.org/?p=28420.

S. E. Seilabi, M. Pourgholamali, G. H. de Almeida Correia, and S. Labi. Robust design of cav-dedicated lanes considering cav demand uncertainty and lane reallocation policy. *Transportation Research Part D: Transport and Environment*, 121:103827, 2023.

N. Sugishita, I. Sevim, M. Carvalho, A. Dems, and R. Atallah. *Fair Network Design Problem: an Application to EV Charging Station Capacity Expansion*. Bureau de Montreal, Université de Montreal, 2025.

D. Z. Wang and H. K. Lo. Global optimum of the linearized network design problem with equilibrium flows. *Transportation Research Part B: Methodological*, 44(4):482–492, 2010.

D. Z. Wang, H. Liu, and W. Szeto. A novel discrete network design problem formulation and its global optimization solution algorithm. *Transportation Research Part E: Logistics and Transportation Review*, 79:213–230, 2015.

J. Wang, X. He, S. Peeta, and W. Wang. Globally convergent line search algorithm with euler-based step size-determination method for continuous network design problem. *Transportation Research Part B: Methodological*, 163:119–144, 2022.

S. Wang, Q. Meng, and H. Yang. Global optimization methods for the discrete network design problem. *Transportation Research Part B: Methodological*, 50:42–60, 2013.

W. Wang, Y. Liu, W. Wei, and L. Wu. A bilevel ev charging station and dc fast charger planning model for highway network considering dynamic traffic demand and user equilibrium. *IEEE Transactions on Smart Grid*, 15(1):714–728, 2023.

J. G. Wardrop. Some theoretical aspects of road traffic research. In *Inst Civil Engineers Proc London/UK/*, 1952.